НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО» Інститут прикладного системного
аналізу Кафедра математичних методів системного аналізу

До захисту допущено:

Завідувач кафедри

_____ Катерина ФАРДМАН

«___»_____20__ р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Системи та методи штучного інтелекту»

спеціальності 122 «Комп'ютерні науки та інформаційні технології»

на тему: «Управління запасами на виробництві на основі удосконаленої методології MRP»

Виконав (-ла):
студент (-ка) IV курсу, групи КА-65
Фардман Катерина Олександрівна _____

Керівник:
Недашковська Надія Іванівна _____

Консультант з економічного розділу:
доцент, к.е.н. Шевчук О. А. _____
Консультант з нормоконтролю:
доцент, к.т.н. Коваленко А.Є. _____
Рецензент:
доцент, к.т.н. Гіоргізова-Гай В. Ш. _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2020 року

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського» Інститут прикладного системного аналізу Кафедра математичних методів системного аналізу

Рівень вищої освіти – перший (бакалаврський)
Спеціальність – 122 «Комп'ютерні науки та інформаційні технології» Освітньо-професійна програма «Системи та методи штучного інтелекту»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Оксана ТИМОЩУК
«25» травня 2020 р.

ЗАВДАННЯ
на дипломну роботу студенту
Фардман Катерині Олександрівні

1. Тема роботи «Управління запасами на виробництві на основі удосконаленої методології MRP», керівник роботи Недашковська Надія Іванівна, асистент, затверджені наказом по університету від
«25» травня 2020 р. № 1143-с

2. Термін подання студентом роботи 8.06.2020.

3. Вихідні дані до роботи

Вихідними даними є програмне забезпечення, що вдосконалює роботу класичної MRP-системи.

Вхідними даними для проведення є результати роботи класичної MRP-системи.

4. Зміст роботи

1. Ознайомитись з методологією MRP та блоком управління запасами.

2. Знайти існуючі методи управління запасами.

3. Вдосконалити блок управління запасами класичної MRP, шляхом додавання існуючих методів.

4. Розробити програмне забезпечення згідно вдосконаленої моделі.

5. Перелік ілюстративного матеріалу (з зазначенням плакатів, презентацій тощо).

5.1. Презентація до захисту роботи.

5.1.1. Слайд «Порівняльна характеристика методів».

5.1.2. Слайд «Побудова удосконаленої методології MRP».

5.1.3. Слайд «Схема удосконаленої системи MRP».

5.1.4. Слайд «Процес аналізу страхового запасу».

6. Консультанти розділів роботи.

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Шевчук О. А., доцент		

7.Дата видачі завдання 01.02.2020

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання	01.02.2020	
2	Збір інформації	15.02.2020	
3	Аналіз вимог завдання, вибір методів і засобів розв’язання поставленої задачі	15.03.2020	
4	Провести пошук та ознайомитися з методологією MRP, а саме блоком управління запасами.	25.03.2020	
5	Ознайомитися з сучасними методами управління запасами.	10.04.2020	
6	Реалізація базового функціоналу блоку управління запасами.	25.04.2020	
7	Додавання знайдених методів	07.05.2020	
8	Оформлення дипломної роботи	31.05.2020	
9	Отримання допуску до захисту та подача роботи в ДЕК	02.06.2020	

Студент Фардман К.О.

Керівник Недашковська Н.І.

РЕФЕРАТ

Дипломна робота: 121 с., 38 рис., 9 табл., 7 додатків, 23 джерел.

ДІАЛОГОВІ СИСТЕМИ, ЧАТ-БОТ, НАВЧАННЯ З ПІДКРІПЛЕННЯМ, ГЛИБОКЕ НАВЧАННЯ, СИМУЛЯЦІЯ КОРИСТУВАЧА, ОБРОБКА ПРИРОДНЬОЇ МОВИ, ІНФОРМАЦІЙНИЙ ПОШУК, БАЗА ЗНАНЬ, ГЛИБИННІ Q-МЕРЕЖІ.

Ця робота присвячена вдосконаленню алгоритму управління запасами на виробництві. На сьогодні логістика підприємства є складною системою багатозадачних процесів. Тому автоматизація логістики є актуальною діяльністю, що вирішує безліч таких проблем як планування роботи, створення та зберігання ресурсів, моделювання стратегій, прогнозування результатів.

Метою роботи є покращення результативності існуючих методів MRP-систем, а також розробка програмного продукту для отримання практичних результатів.

Об’єктом дослідження є сучасні методи управління запасами, їх алгоритми та процеси.

ABSTRACT

The work consist of 121 pages 38 images 9 tables 23 sources

The theme: «A deep reinforcement learning chatbot».

DIALOGUE SYSTEMS, CHATBOT, REINFORCEMENT LEARNING, DEEP LEARNING, USER SIMULATION, NATURAL LANGUAGE PROCESSING, INFORMATION SEARCH, KNOWLEDGE BASE, DEEP Q-NETWORK.

This work is dedicated to improving the inventory management algorithm in production. Today, enterprise logistics is a complex system of multitasking processes. Therefore, automation of logistics is an actual activity that solves many problems such as scheduling, creating and storing resources, modeling strategies, forecasting results.

The purpose of the work is to improve the performance of existing methods of MRP-systems, as well as to develop a software product for practical results.

The subject of the study is modern inventory management methods, their algorithms and processes

ЗМІСТ

ВСТУП

Управління запасами на виробництві є у першу чергу питанням логістики. MRP-система є логістичною системою, що реалізує системний підхід до вирішення проблем. Вона належить до мікрологістичних систем, що вивчає локальні проблеми управління матеріальними та інформаційними потоками на внутрішньому рівні. У той час як макрологістичні системи розглядають глобальні проблеми.

Формування та організація функціонування сучасної логістичної системи неможливі без інформаційних технологій. Саме завдяки розвитку інформаційних систем та технологій, який забезпечує автоматизацію технологічних операцій та прийняття раціональних управлінських рішень в режимі реального часу, логістика стала домінуючою формою організації товароруху на ринках економічно розвинутих країн. З цих позицій логістика повинна будуватись на базі сучасних інформаційних систем та технологій:

- технологій управління та моделювання логістичних бізнес-процесів CALS I CASE;
- електронного документообігу (EDI-технологій); інтернет-рішень, мобільного та електронного бізнесу; систем сканування штрих-кодів та радіочастотної ідентифікації вантажів (RFID);
- голосової технології комплектування товарів (Pick-by-Voice); супутникових систем зв'язку і навігації, що дозволяють відстежувати товарно-транспортні потоки.

Ціль логістичної системи досягається за рахунок внутрішньої та зовнішньої координації дій її компонентів. Тому для ефективного її функціонування треба створити механізм миттєвої координації стратегій

виробництва. Виділяють два напрямки досягнення необхідного рівня координації в управлінні логістичними потоками :

1. посилення взаємодії між різними функціональними ланками;
2. організаційні перетворення в структурі підприємства.

На практиці ці напрямки доповнюють один одного, при цьому використовуються різні методи координації за допомогою розроблених процедур, які регламентують дії менеджерів з управління потоками (посадові інструкції, нормативні документи, що визначають завдання, повноваження і послідовність дій керівників різних функціональних служб та їх підлеглих з управління матеріальними ресурсами і запасами на різних етапах їх руху). Також з цією метою широко використовуються спеціалізовані інформаційні системи, що дозволяють оперативно погоджувати плани постачання, виробництва та збуту у довгостроковій та короткостроковій перспективі та забезпечувати збалансоване поточне регулювання і контроль матеріальних та інших ресурсів з урахуванням змін.

Особливою сферою функціонування логістичної системи є інформаційна підсистема. Об'єктом логістичної інформаційної системи (інформаційної логістики) є інформаційний потік. Управління цим потоком може стосуватись виконання операційних логістичних функцій (зберігання, передавання, обробка інформації) і виконання стратегічних завдань логістичної системи (рішень щодо учасників логістичного ланцюга, вибір технологій тощо). Логістичні інформаційні системи можуть створюватись з метою управління потоками як на рівні підприємства (мікрорівні), так і на макрорівні (регіону, країни тощо).

РОЗДІЛ 1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Актуальність задачі

У нашому глобалізованому світі з купою месенджерів, task-менеджерів та ERP-систем, на підприємстві все більш гострою стає проблема конфліктів між різними відділами. Кожен з них має свої задачі, показники ефективності, цілі, інструменти та ресурси, однак, критично залежить від інших. Це значно уповільнює розвиток бізнесу, демотивує працівників, приносить збитки.

Конфлікти між функціональними підрозділами підприємства трапляються через динамічність всієї системи: асортимент збільшується, його життєвий цикл зменшується, з’являються нові ринки збуту, змінюється ринок і так далі.

Спрогнозувати вплив найменшої зміни хоч одного з цих аспектів – надскладна задача. Через це досягнення або обмеження одного підрозділу суперечать з роботою іншого. Тому досягаючи оптимуму в одному відділі підприємства чи навіть в декількох, ми не гарантуємо собі підвищення спільних показників.

Наприклад, відділ планування може потерпати від поганого прогнозу продажів, бо з’явився неочікуваний великий контракт. Відділ продажу в свою чергу не може спрогнозувати продажі нового асортименту, та появи нового каналу збуту, що забезпечив маркетинг. Отже, невеличка зміна в одному місці провокує проблеми в багатьох інших. Всі ці проблеми підприємства намагаються вирішити за допомогою сучасних MRP-систем або модулів. Проте їх складність змушує працівників використовувати систему лише як базу даних. Отже, актуальною задачею на сьогодні є спрощення використання MRP-систем та удосконалення MRP-алгоритму.

1.2 Аналіз методології MRP

1.2.1 MRP-системи та принципи їх роботи

У сучасному світі майже кожне підприємство будує власну MRP-систему для планування використання ресурсів. Причому цей план не повинен мати лишки або втрачені продажі. Цілями MRP-системи є:

- Безперебійна робота підприємства;
- Мінімізація запасів на складах;
- Регулювання поставок матеріалів; [14]

MRP-системи належать до мікрологістичних. Мікрологістичні системи є підсистемами, структурними складовими макрологістичних систем. До них відносять різні виробничі і торговельні підприємства, територіально-виробничі комплекси.[2] Мікрологістичні системи являють собою клас внутрішньовиробничих логістичних систем, до складу яких входять технологічно пов'язані виробництва, об'єднані єдиною інфраструктурою. В рамках макрологістики зв'язки між окремими мікрологістичними системами встановлюються на базі товарно-грошових відносин. Усередині мікрологістичної системи також функціонують підсистеми. Однак основа їх взаємодії безтоварна. Це окремі підрозділи всередині фірми, об'єднання або іншої господарської системи, що працюють на єдиний економічний результат. [6] Успіх системи MRP здебільш побудований на своєчасному отриманні інформації про потреби ринку, актуальності інформації про запаси матеріалів, організації виробничого плану, а також створення плану закупівлі компонентів. Ці аспекти, безперечно, мають під собою безліч факторів та критеріїв, що можуть критично вплинути на систему. Саме це робить її надскладною в створенні, впровадженні та регулюванні. Однак чітка робота MRP-системи здатна принести

підприємству багато позитивних ефектів. Наприклад, якщо вдасться зменшити кількість запасів на складі, підприємство зможе вивести так звані «заморожені» кошти, які воно витрачає кожного разу на матеріали, що не використовуються в даний момент. А це, в свою чергу, дасть змогу витратити гроші на розвиток або оптимізацію виробництва. Чітке регулювання поставок матеріалів зекономить гроші на утримання ресурсів на складі. Безперебійна робота підприємства забезпечить зменшення браку та підвищить чіткість планування програми виробництва.

Інформаційні логістичні системи повинні задовольняти наступним вимогам: масштабованість, распределенность, модульність, відкритість. [3]

Масштабованість - здатність системи підтримувати у вигляді окремих користувачів, так і безліч користувачів.

Розподіленість - здатність системи забезпечувати спільну обробку документів кількома територіально рознесеними підрозділами підприємства або кількома віддаленими одна від одної робочими місцями.

Модульність - здатність системи надавати користувачам можливість налаштовувати і вибирати функції системи, виходячи із специфіки і складності діяльності підприємства, тобто система автоматизації – гнучка і складається з окремих модулів, інтегрованих між собою (збут, склад, закупівлі, виробництво, персонал, фінанси, транспорт).

Відкритість - система автоматизації інтегрована в інші інформаційні системи, вона має відкриті інтерфейси для розробки нових додатків і інтеграції з іншими системами.

Розглянемо детальніше алгоритм MRP. Задля чіткого планування логістики ресурсів на підприємстві, кожен матеріал обов’язково має один з чотирьох статусів, що відповідають рівню готовності i-того матеріалу для виробничого процесу:

- Наявний на складі (W_i)
- Зарезервований (R_i)

- В поточних замовленнях (O_i)
- Планується замовлення (P_i)

Також передбачається що для кожного матеріалу є свій страховий запас (S_i), що визначається для кожного окремого випадку індивідуально.

Алгоритм MRP працює за принципом обчислення повної та чистої потреби в матеріалах, та постійного поновлення вже діючих планів на основі цієї інформації. Повною потребою (N_i^t) називається розрахунок всіх необхідних ресурсів та матеріалів для якогось проміжку часу. Чистою потребою (N_i^p) називається розрахунок ресурсів або матеріалів, що треба замовити або виготовити протягом якогось проміжку часу.

$$N_i^p = N_i^t - W_i - S_i - R_i \quad (1.1)$$

Таким чином, щойно чиста потреба стає більше за нуль, MRP-система автоматично формує нове замовлення. Задачею MRP-системи є розробка виробничого плану, що повністю задовольняє чисту потребу. [5]

Отже схематично роботу MRP-системи можна представити на рисунку 1.1.

Рисунок 1.1 Схема роботи класичної MRP-системи.

На вході система отримує звіти про прогнозований попит, наявні запаси та діючі замовлення, специфікацію виробничих продуктів на визначений проміжок часу. За допомогою алгоритму MRP розраховуються потреби підприємства, що формують новий виробничий план та план закупівлі матеріалів. На рисунку 1.2 схематично зображено структуру модуля формування замовлень.

Класична система MRP-II (Standart System) містить опис 16 груп функцій системи:

1) Sales and Operation Planning (планування продажів і виробництва);
2) Demand Management (управління попитом);
3) Master Production Scheduling (складання плану виробництва);
4) Material Requirement Planning (планування матеріальних потреб);
5) Bill of Materials (специфікації продуктів);
6) Inventory Transaction Subsystem (управління складом);
7) Scheduled Receipts Subsystem (планові поставки);
8) Shop Flow Control (управління на рівні виробничого цеху);
9) Capacity Requirement Planning (планування виробничих потужностей);
10) Input / output control (контроль входу / виходу);
11) Purchasing (матеріально технічне постачання);
12) Distribution Resourse Planning (планування ресурсів розподілу);
13) Tooling Planning and Control (планування та контроль виробничих операцій);
14) Financial Planning (управління фінансами);
15) Simulation (моделювання);
16) Performance Measurement (оцінка результатів діяльності). [4]

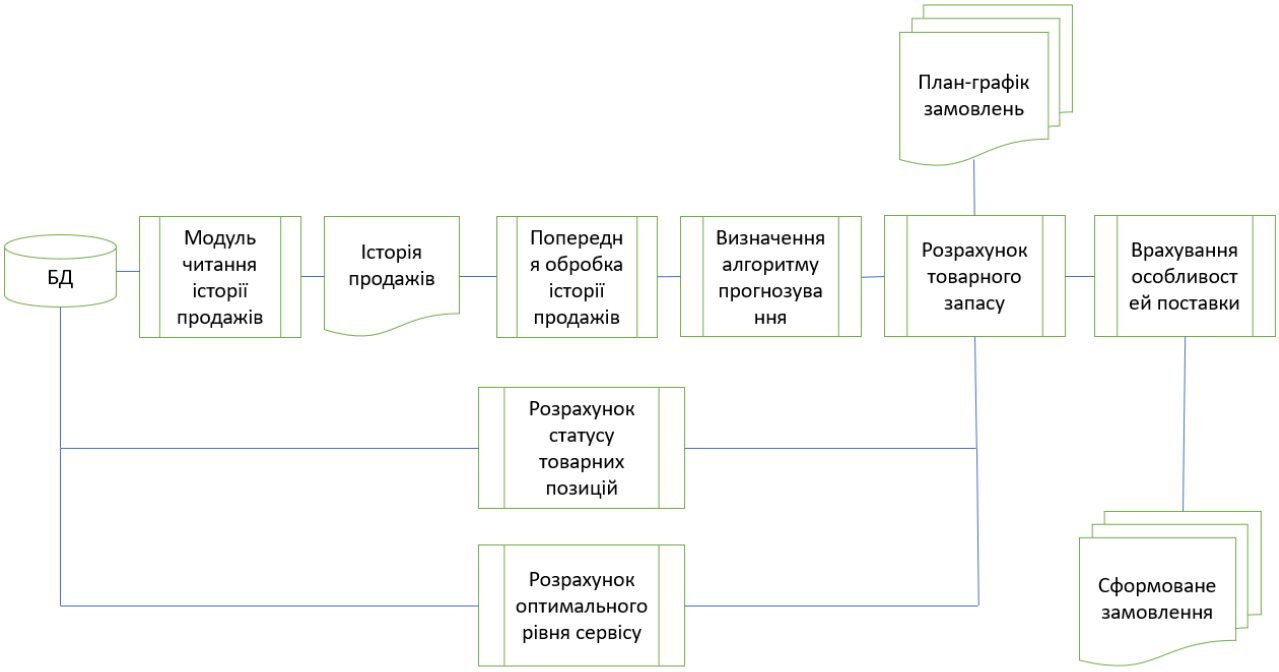


Рисунок 1.2 Структура модуля формування замовлень.

1.2.2. Управління ланцюгами поставок

Ланцюгом поставок є сукупність організацій (від джерела ресурсів до споживача), що взаємодіють між собою у матеріальних, фінансових та інформаційних потоках. Ланцюг поставок схематично зображено на рисунку 1.3.

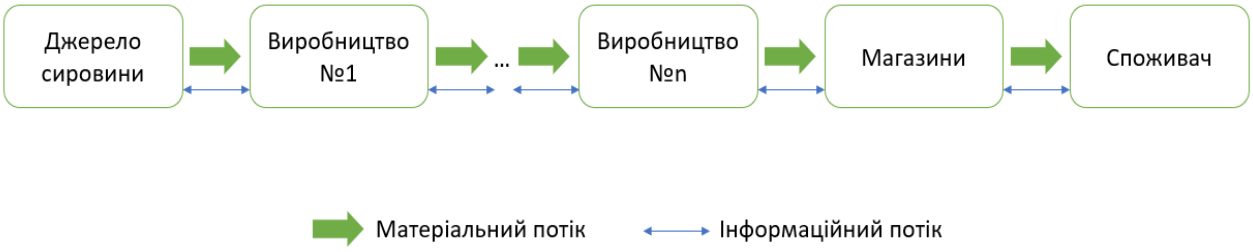


Рисунок 1.3. Схема ланцюга поставок.

Матеріальний потік, що крокує ланцюгом поставок, проходить через ряд ланок. Управління матеріальним потоком на цьому етапі має свою специфіку та відноситься до виробничої логістики.

Історично склалось так, що управління ланцюгами поставок виконується двома способами: виштовхуванням або витягуванням.

Виштовхування, до якого належить класична MRP-система, засноване на чіткому прогнозі та специфікаціях. При прогнозуванні враховуються норми витрат, коефіцієнти переходу та інше. Таким чином системи здатні об'єднати у собі контроль усього виробничого процесу, як на рисунку 1.4. Зростаючі масштаби підприємства та виробництва постійно потребують вдосконалення програмного, інформаційного та технічного забезпечення системи. Параметри матеріального потоку наближаються до оптимуму прямо пропорційно здатності керуючої системи знайти та оцінити усі фактори впливу на виробничу ситуацію. При цьому страховий запас, що формується наперед, посилює так званий «ефект батога».

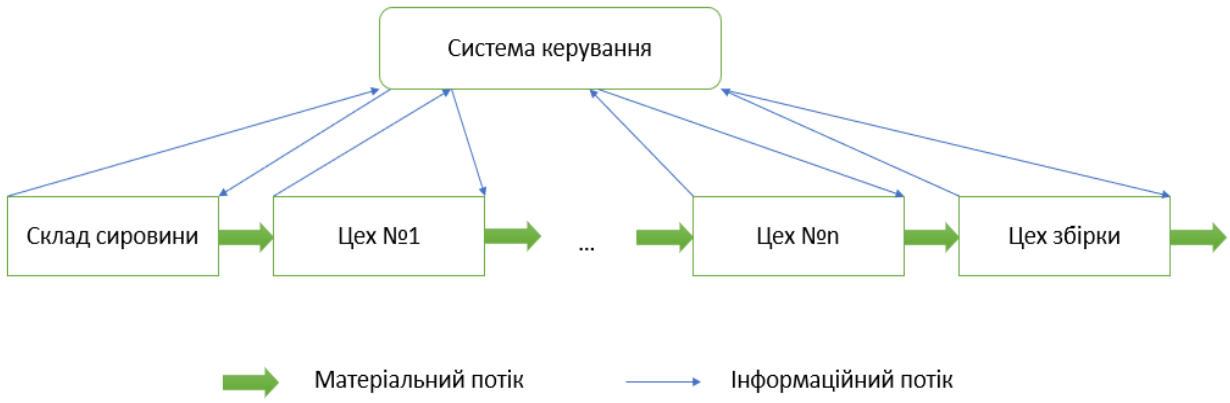


Рисунок 1.4. Виштовхувальна система керування ланцюгом поставок.

У 70-х роках з'явився новий підхід, що на відмінну від виштовхування пропонував створювати продукцію тільки під замовлення, тому і називався протилежно – витягування. Витягуюча система керування ланцюгом поставок зображена на рисунку 1.5.

До методів витягування належать lean (ощадливе виробництво), теорія обмежень систем (TOC) та інші. Методи витягування не концентруються на підвищенні точності прогнозу, а збільшують гнучкість усіх процесів виробництва задля швидкої та безболісної реакції на зміни.[15] В цьому виробничому процесі матеріали передаються по ланкам тільки за необхідності.

Система керування не втручається у кожний етап, а взаємодіє тільки з останньою ланкою технологічного ланцюга, ставлячи йому завдання. Виробнича програма кожної ланки залежить від замовлення наступної.

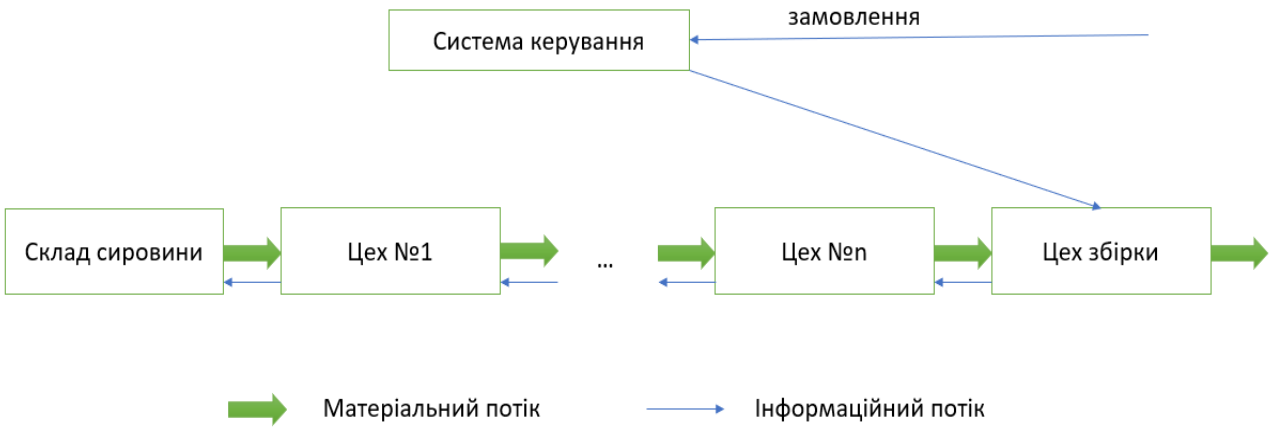


Рисунок 1.5 Витягуюча система керування ланцюгом поставок.

Таке рішення було створене через те, що персонал на окремих ланках виробничого ланцюга здатен виявити та врахувати значно більше факторів ніж центральна керуюча система. За допомогою такої дискретності на кожному етапі виробництва максимізується ефективність.

Однак при появі нового асортименту, сильної сезонності або частіших промоактивностей навіть ці системи породжують відхилення та руйнуються.

1.3 Эффект батога

Система планування матеріалів насправді пов’язана і з деякими складнощами. По-перше, система є дискретним динамічним мультиагентним середовищем, що означає, що будь-яка зміна плану, є причиною для миттєвого перепланування усього виробничого процесу. Чим більшим є масштаб підприємства, тим більш складним та великим є об’єм обчислень та обробки даних. Це потребує чіткої, швидкої та безпомилкової роботи професіонала. [1]

По-друге, підприємство за допомогою MRP-системи прагне зменшити запаси на складах. Через це зростають логістичні витрати на транспортування, зберігання матеріальних ресурсів та готових продуктів.

По-третє, нажаль такі системи не є чутливими до короткочасної зміни попиту ринку. Чіткість планування сильно залежить від достовірності інформації, а також швидкості реагування підприємства. А короткочасні зміни попиту важко піддаються прогнозуванню. Цей аспект викликає найбільшу слабкість MRP-систем: ефект батога.

Ефект батога – феномен в ланцюгах поставок, який полягає в посиленні амплітуди коливання попиту (обсягу замовлень) в міру віддалення від реального джерела попиту в ланцюзі постачань.[7] Він виникає через те, що замовлення та матеріали передаються по ланцюгу поставок в обох напрямках з затримками в часі. Чим більше ланок у ланцюзі поставок і чим тривалішим є час виконання замовлень, тим більшою є ця амплітуда.

В наслідок малого коливання попиту на початку ланцюга поставок спостерігається сильне коливання окремо взятого продукту в кінці ланцюга, як на рисунку 1.6. Таким чином запаси на складах можуть швидко трансформуватися або в стан дефіциту або в надлишки.

Розглянемо графічно вплив ефекту батога:

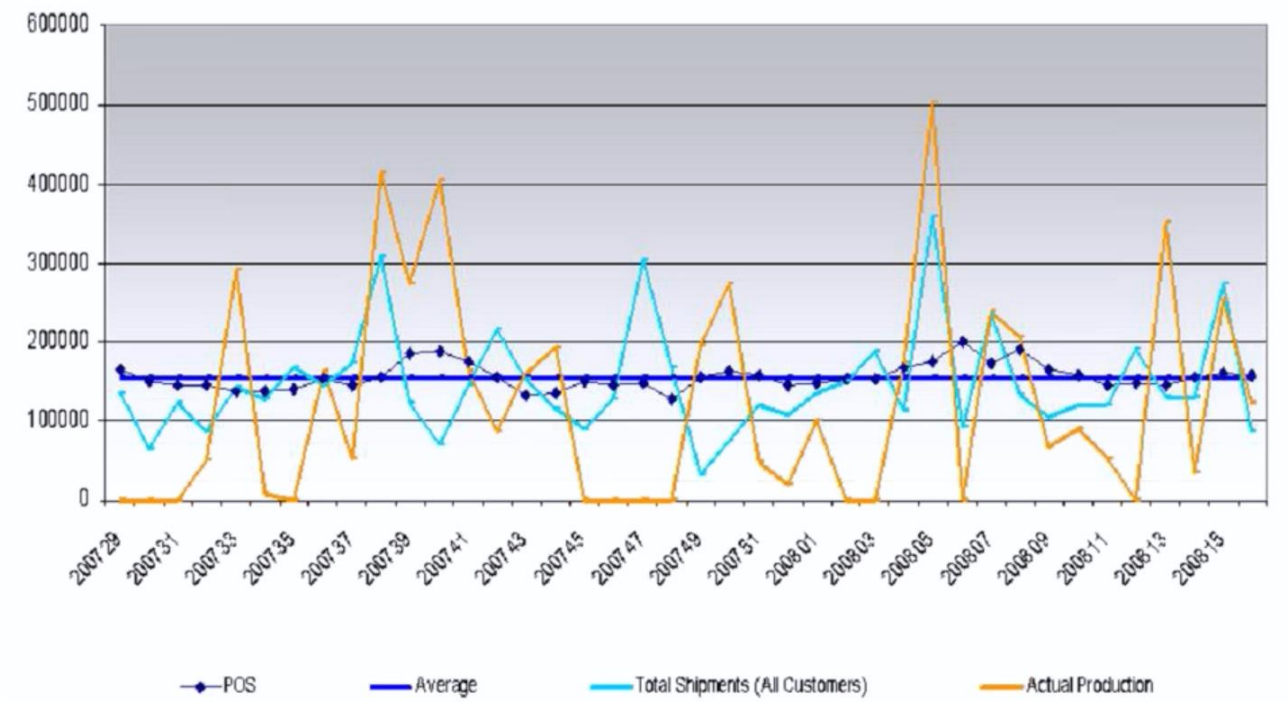


Рисунок 1.6 Вплив ефекту батога у класичній MRP-системі.

На графіку можна побачити коливання об’єму продажів (POS), коливання

об'єму відвантаження за замовленнями (Total Shipments), коливання об'єму виробництва (Actual Production). Графік також виявляє погані наслідки цього ефекту (втрачені продажі, затримки замовлень, надлишки продукці). Об’єм продажу за заданий період часу стрибав не більше ніж на 25% від середніх показників. Це потягнуло за собою стрибки об’ємів відвантаження за замовленнями на 105%, а об’ємів виробництва на 150%.

Х. Лі і співавт. (1997) запропонував основні правила для зниження негативного ефекту батога: обмін інформацією, координація в каналі просування, збільшення операційної ефективності. Дана робота була піддана численній критиці і з тих пір багато авторів звернулися до більш конкретних рішень і інструментів:

- Політика розміщення замовлень в ланцюзі постачань
- Правила обсягу замовлення
- поліпшення прогнозування
- Зниження волатильності попиту
- Багато-Агентно підхід

Обмін інформацією є одним з найбільш важливих інструментів для мінімізації ефекту батога. Більшість сучасних інструментів і підходів, в тому числі VMI, CPFR, і т.д. використовують цей принцип. Важливість інформації в ланцюзі постачань:

- Сприяє зниженню мінливості в ланцюгах поставок
- Допомога постачальникам в поліпшенні прогнозів
- Дозволяє координувати системи і стратегії виробництва і розподілу
- Дозволяє роздрібної торгівлі краще обслуговувати своїх клієнтів
- Дозволяє роздрібної торгівлі швидше реагувати і адаптуватися до проблем ланцюга поставок
- Дозволяє скоротити час виконання замовлень

1.4 Висновки

Концепція MRP-систем була сформульована у кінці 1960-тих років Олівером Уайтом. Він описав підхід, що сумує управління логістикою на виробництві та застосування обчислювальної техніки. І це вперше дозволило оперативно коригувати планові завдання в процесі виробництва (при зміні потреб, коригування замовлень, нестачі ресурсів, відмовах обладнання).

MRP-системи насьогодні є невід'ємним інструментом управління та логістики на виробництві. За її допомогою можна швидко та якісно вирішити питання управління запасами, прогнозування попиту та продажів, створення листу закупівлі, планування виробничих процесів та навіть моделювання. Система потребує постійного вдосконалення її забезпечення та постійного корегування вхідних даних, однак, швидко будує плани, що раніше займали тиждень, та економить заморожені кошти підприємства. На відміну від методів теорії управління запасами, які передбачають незалежний попит на всю номенклатуру, MRP часто називають методом розрахунків для номенклатури «Залежного попиту» (тобто формування замовлень на вузли і комплектуючі вироби в залежності від замовлення на готову продукцію).

Класичні MRP-системи так як і всі системи виштовхування зараз є застарілими, та придатні для досить вузької сфери діяльності з списком обмежень. Вони потребують вдосконалення, шляхом перебудови основних методологічних процесів. Більш нові їх родичі не можуть встояти перед ефектом батога. Метою даної роботи є покращення алгоритму MRP шляхом додавання ознак сучасних методів витягування та прибирання недоліків.

РОЗДІЛ 2 ВИБІР І ОПИС МАТЕМАТИЧНИХ МОДЕЛЕЙ СУЧАСНИХ МЕТОДІВ

2.1 Вибір методів для порівняння

У сучасному менеджменті відомо, як мінімум, кілька високоефективних і перевірених на практиці систем, застосування яких веде до істотного зростання ефективності ведення бізнесу, поліпшення виробництва, збільшення обсягів продажів і т.д. Іноді вони конкурують один з одним. Наприклад, не всім приходить в голову, що правила японського менеджменту часом непогано поєднуються з західноєвропейським чи американським. Таке поєднання призводить до більш високих результатів за рахунок ефекту синергії.

Дослідження, нещодавно проведені на підприємствах Global Electronics Manufacturer, показали, що кращих результатів вдається домагатися керівникам тих заводів, де застосовується відразу декількох систем: Lean, SixSigma і TOC. Відсоток внеску в економіку від підприємств, що використовують Lean – 4%, SixSigma – 7%, Lean &SixSigma &TOC – 89%.

TOC застосовують, коли необхідно визначити зони уваги. Обмеження шукають в першу чергу. Lean використовують для прискорення виконання замовлень і підвищення ефективності виробництва. 6 Sigma зазвичай сприймають як методику поліпшення якості.

2.2 Опис теорії обмеження систем

Теорія обмеження систем була сформульована 80-их роках ізраїльським вченим Голдратом. Її філософія полягає в тому, що ефективність системи залежить від продуктивності обмеження. За принципом теорії слід постійно шукати обмеження в системі, потім підлаштовувати систему під нього, розширювати це обмеження і знову шукати нове обмеження системи.[11] Розглянемо ланцюг поставок з заданими обмеженнями пропускної здатності (рисунок 2.1):

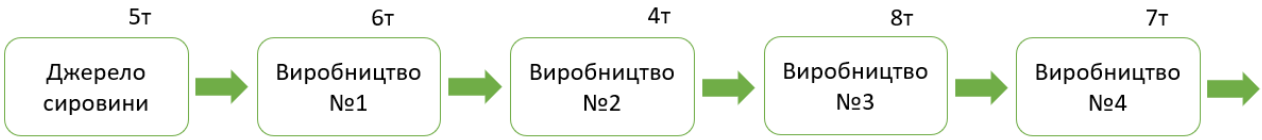


Рисунок 2.1 Ланцюг поставок.

Джерело сировини випускає 5т сировини в відрізок часу, Виробництво№1 здатне обробити 6т сировини, та передати її далі, Виробництво №2 здатне обробити 4т цього ресурсу і т.д. Коли постає питання, що саме треба вдосконалювати в цій системі, теорія обмежень говорить – ланку з найменшою пропускною здатністю, тобто Виробництво №2. Далі підвищення ефективності системи проходить через два етапи. Спочатку треба підпорядкувати цьому обмеженню усю систему. В теорії зрозуміло, що уся система матиме на виході 4т продукції. Але на практиці через брак виробництва, зламане обладнання, запізнення на роботу працівників матимемо 2-3т продукції. Тож першим кроком варто максимізувати ефективність ланки ланцюга, що його обмежує: найняти ще працівників, відремонтувати обладнання і т.д, щоб отримувати гарантовані 4т продукції. Другим кроком вже буде розширення цього обмеження: закупівля нового обладнання, розширення цеху. Коли обмеження подолане, починається пошук нового обмеження системи.

Операційний рівень теорії обмеження систем містить три показники:

1. «Прохід» - швидкість, з якою система генерує гроші в результаті продажів.

2. Товарно-матеріальні цінності – гроші, які розширюють Прохід, тобто «…загальна сума грошей, інвестована системою у купівлю того, що вона намагається продати у кінцевому рахунку». []
3. Операційні витрати – загальна сума грошей, витрачена системою на переведення товарно-матеріальних цінностей у прохід.

Концепція ТОС полягає у скороченні операційних витрат і товарно-матеріальних цінностей, та збільшення проходу.

У теорії обмежень також є найпопулярніша методика «Барабан – Мотузка–Буфер», що зображена на рисунку 2.2.

Барабан – це ланка ланцюга з найбільш вузьким виробничим ресурсом.

Мотузка – це принцип «зв'язування» попередніх ланок до барабану з тактом барабану. Тобто витягування рівно стількох ресурсів, скільки потрібно для безперебійної роботи барабану. Схема роботи мотузки зображена на рисунку 2.3.

Буфер – це час випередження запуску матеріалів у виробництво від часу обробки буфером матеріалів.

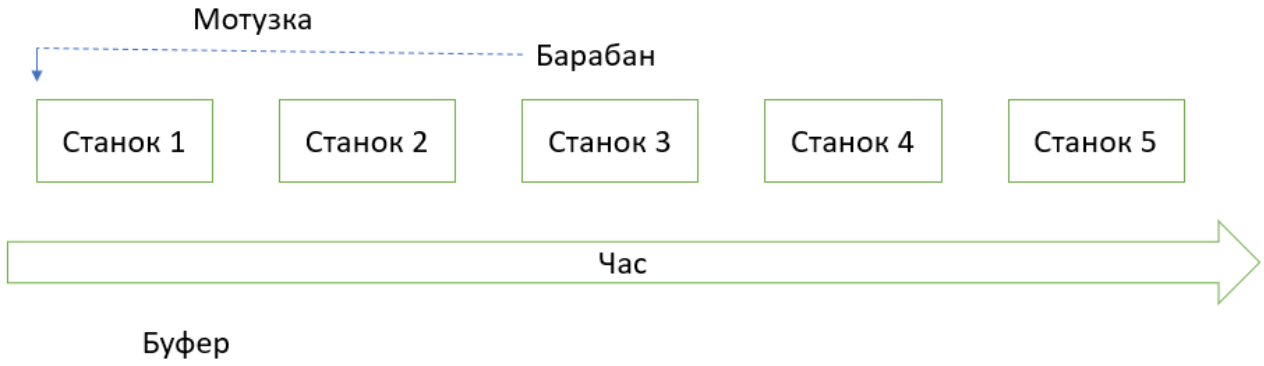


Рисунок 2.2 Схема принципу роботи методики «Барабан – Мотузка – Буфер»

Розглянемо як за допомогою ТОС можна підвищити ефективність управління бізнес-процесами. Нехай процес являє собою напрямлений граф $P = (\Omega, A, L, U, R)$, де:

Ω – швидкість появи екземплярів процесів.

$A = \{a_i | i = 1, \dots, I\}$ – набір дій, які входять у екземпляр процесу. a_i – і-та дія, а I – загальна кількість дій у графі P.

$L \subseteq \{(a_i, a_j) \mid a_i \in A, a_j \in A \,\&\, i \neq j\}$ – набір зв'язків, де (a_i, a_j) означає, що дія a_i виконується перед a_j.

$U = \{u_k | k = 1, \dots, K\}$ – набір учасників процесу, де К – загальна кількість учасників.

$R(\subseteq A \times U) = \{(a_i, u_k) | a_i \in A, u_k \in U\}$ – набір взаємозв'язків, де u_k учасник виконує a_i дію.

За методологією теорії обмеження систем нам потрібно знайти учасника, що є критичним ресурсом системи, тобто найбільше її обмежує. Ефективність учасника залежить від того який об'єм задач він має. Ми повинні оцінити пріоритетність навантаження учасника у різних процесах. Для цього запозичимо апарат теорії масового обслуговування [13]. Розрахуємо навантаження учасника WL_k.

$$WL_k = \sum_{\{a_i | (a_i, u_k) \in R\}} \frac{\lambda_{a_i} \times p_{a_i u_k}}{\mu_{a_i u_k}} = \Omega \times \sum_{\{a_i | (a_i, u_k) \in R\}} \frac{f_{a_i} \times p_{a_i u_k}}{\mu_{a_i u_k}}, \forall u_k \in U \qquad (2.1)$$

Де:

f_{a_i} – очікуваний час виконання задачі по дії a_i.

λ_{a_i} – швидкість появи задачі по дії a_i.

$p_{a_i u_k}$ – ймовірність розподілення задачі по дії a_i учаснику u_k.

$\mu_{a_i u_k}$- середній час виконання учасником u_k задачі по дії a_i

Використовуючи цю формулу, можна знайти учасника у котрого найбільший пул задач. А це значить, що треба контролювати його навантаження. [20]

$$WL_{CR} = \max WL_k \qquad (2.2)$$

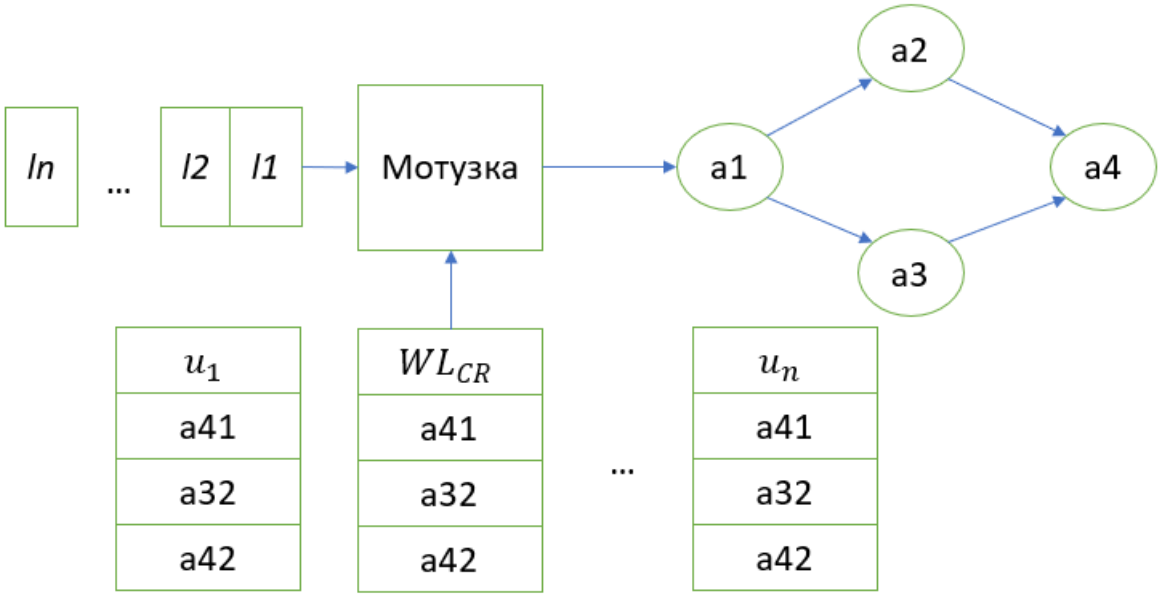


Рисунок 2.3 Схема роботи інструменту «Мотузка» у методиці «Барабан – Мотузка – Буфер».

Проведемо аналогії з інструментами теорії обмежень:

Барабан – це час за який критичний ресурс виконує задачі. Допустимо, що критичний ресурс – це команда, котра виконує одну задачу a_i за очікуваний час ET_i. Тоді барабан можна розрахувати як:

$$Drum(d) = \sum_{\{a_i|(a_i,u_k)\ \in\ R\}} \frac{1}{ET_i} \tag{2.3}$$

Буфер – інші учасники, які можуть попередити перенавантаження або простій критичного ресурсу.

Мотузка – поява екземплярів процесів у згоді з пропускною можливістю критичного ресурсу.

$$\lambda = d\left(\frac{\text{задача}}{\text{час}}\right) \tag{2.4}$$

Не дивлячись на те, що теорія існує вже багато років, вона не стала прикладним інструментом сучасних підприємців. Причинами цього є:

- Надто широкі горизонти застосування, через універсальність теорії.
- Великі витрати часу на створення вже існуючих процесів, через те що виконавець сам повинен придумати спосіб розширення (вдосконалення) обмеження.
- Важке впровадження підходу ТОС, у тому числі і на виробництві.

2.3 Опис методології lean (ощадливе виробництво)

Ідея ощадливого виробництва народилась у виробничій системі японської компанії Toyota. В часи, коли ця компанія випускала низькоякісні автомобілі, було вирішено не тільки мінімізувати витрати на процеси, що не діють на кінцевого споживача, але й робити це безперервно. Покупця автомобіля зовсім не цікавить як довго запчастини лежали на складі, та у яких умовах, або скільки разів їх перевозили з місця на місце. Завдяки цьому відбувся феномен, що допоміг не великій японській компанії стрибнути у якості, не підвищуючи ціну за автомобіль. Це сильно потіснило американський автопром на ринку.

Ощадливе виробництво як концепція набуло сили передбачаючи залучення до процесу оптимізації бізнесу кожного співробітника і максимальну орієнтацію на споживача. Відправна точка концепції - оцінка цінності продукту для кінцевого споживача, на кожному етапі його створення. В якості основного завдання передбачається створення процесу безперервного усунення втрат, тобто усунення будь-яких дій, які споживають ресурси, але не створюють цінності (не є важливими) для кінцевого споживача.

Відповідно до концепції бережливого виробництва, вся діяльність

підприємства поділяється на операції та процеси, що додають цінність для споживача, і операції і процеси, що не додають цінності для споживача. Завданням «бережливого виробництва» є планомірне скорочення процесів і операцій, що не додають цінності. Реалізація концепції передбачає застосування таких підходів, як «точно у час» і принципів витягування виробництва.

Ощадливе виробництво формулює види витрат, що підлягають мінімізації, вони наведені у таблиці 7.1

Таблиця 7.1 Порівняння lean-витрат та витрат виробництва

Lean-витрати	Витрати виробництва
витрати через перевиробництво	Постійні витрати
витрати часу через очікування	Змінні витрати
витрати при непотрібному транспортуванні	Валові витрати
витрати зайвих етапів обробки	Середні витрати
витрати через зайві запаси	Витрати на одиницю продукції
витрати за непотрібні переміщення	Прямі витрати
витрати через випуск дефектної продукції	Поточні витрати
нереалізований творчий потенціал співробітників	Граничні витрати
перевантаження робочих	Альтернативні витрати
нерівномірність виконання операції	Витрати майбутніх періодів

Ощадливе виробництво як процес включає в себе п'ять етапів:

1. Визначення цінності конкретного продукту.
2. Визначення потоку створення цінності для цього продукту.
3. Забезпечення безперервного потоку створення цінності продукту.
4. Надання можливості споживачеві отримувати продукт.
5. Прагнення досконалості.

Складемо математичну модель для опису мінімізації витрат по lean:

Нехай $C_{заг}$ – загальний рівень витрат, $C_{зам}$ – сума витрат на розміщення замовлення, $C_{зб}$ – витрати на зберігання сировини та матеріалів, К – вартість розміщення одного замовлення, N^t – повна потреба, N^p – чиста потреба. За методологією нам потрібно мінімізувати загальний рівень витрат виробництва:

$$C_{заг} = C_{зам} + C_{зб} \rightarrow \min \tag{2.5}$$

$$C_{зам} = K \cdot \frac{N^t}{N^p} \tag{2.6}$$

$$C_{зб} = C_{збо} \cdot \frac{N^p}{2} \tag{2.7}$$

Додамо вартість запасів $C_{зап}$, щоб оцінити її залежність від об’єму замовлення:

$$C_{заг} = C_{зам} + C_{зб} + C_{зап} \rightarrow min \tag{2.8}$$

За формулою Уилсона оптимальний об’єм партії замовлення сировини та матеріалів має вигляд:

$$Q_o = \sqrt{\frac{2 \cdot N^t \cdot K}{C_{збо}}} \tag{2.9}$$

Для збільшення точності введемо функціональні залежності рівня витрат від об’єму замовлення:

$K(Q)$ – функція залежності вартості розміщення замовлення від його об’єму.

$C_{збо}(Q)$ – функція залежності вартості зберігання одиниці запасів від їх об’єму.

$C_{запо}(Q)$ – функція залежності вартості одиниці запасів від об'єму замовлення.

Отже функція для оптимізації приймає наступний вигляд:

$$C_{заг} = K(Q) \cdot \frac{N^t}{N^p} + C_{збо}(Q) \cdot \frac{N^p}{2} + C_{запо}(Q) \cdot N^t \rightarrow min \tag{2.10}$$

[21]

Ощадлива логістика (lean-логістика) - витягаюча система логістики, яка об'єднує весь ланцюг постачальників, задіяних в потоці створення цінності, в якій відбувається часткове поповнення запасів невеликими партіями, основний показник такої системи - сукупна логістична вартість.

Логістична технологія LP (Lean production) є розвитком технології "точно у час". Суть технології LP полягає в поєднанні таких логістичних компонентів, як висока якість, дрібний розмір виробничих партій, низький рівень запасів, висококваліфікований персонал, гнучке устаткування. У цій технології з'єднані переваги масового (низька собівартість виробництва) і дрібносерійного (розмаїтість продукції та широкий асортимент) виробництв, що дозволяє досягти високої якості продукції, низьких виробничих витрат, швидкої реакції на споживчий попит, оперативної переналагодження устаткування. Опорними елементами логістичного процесу в технології LP є:

- скорочення підготовчо-заключного часу;

- зменшення розмірів партій продукції;

- скорочення основного виробничого часу;

- контроль якості всіх процесів;

- скорочення логістичних витрат у виробництві;

- наявність надійних постачальників;

- еластичні потокові процеси;

- витягує принцип організації системи.

Для Lean production тягнуча система означає мінімізацію запасів па полицях, розміщення практично всіх запасів на робочих місцях, тобто використання лише тих компонентів бізнес-процесу, які необхідні для задоволення конкретного замовлення споживача.

Слід виділити два напрями інтеграційного процесу в логістиці:

1. вертикальна інтеграція - відповідає процесам злиття, поглинання, укрупнення підприємств, спрямована па підвищення їх ефективності

шляхом об'єднання виробництв суміжних галузей, пов'язаних між собою по технологічному ланцюжку "постачальник - виробник - споживач";

2. горизонтальна інтеграція - відповідає об'єднання господарюючих суб'єктів одного профілю. До них, зокрема, відносяться мережні форми горизонтальної інтеграції, роздрібні торгові мережі, стратегічні альянси компаній, які об'єднують свою діяльність для реалізації спільних стратегічних цілей у створенні логістичної інфраструктури, інформаційного забезпечення і т. п.

Логістична інтеграція торкається ще один аспект - формування логістичного потоку. Його субстанціональний склад включає матеріальні, інформаційні, фінансові потоки, починає активно освоюватися четвертий потік - кадровий, досліджуються і робляться спроби визначити правовий та сервісний потоки.

2.4 Висновки

Ми розглянули три методології витягування засновані на різних філософіях. Одна з них заснована на знаходженні та розширенні «вузького місця», інша на мінімізації витрат на бездохідні процеси, третя у постійному вдосконаленні існуючих процесів. У них є свої переваги та недоліки, однак у їх синергії, що має назву TLS, є дуже багато прихильників.

TLS вбирає в себе усі сильні сторони трьох методів, та відсікає недоліки. До того ж виявилось, що вони чудово доповнюють один одного як показано у таблиці 2.1.

Таблиця 2.1. Порівняльна характеристика методів.

Теорія обмежень	- Зосереджується на покращенні обмежень, що визначають загальну продуктивність системи. - Значно підвищує рентабельність інвестицій та успіх програм Lean&SixSigma - Збільшує прибуток за рахунок збільшення продажів, а не за рахунок скорочення витрат, а отже.
Ощадливе виробництво	- Найрозповсюдженіший підхід у виробництві по всьому світу. - Сфокусований на знаходженні та зменшенні усіх форм витрат - Багатовимірний підхід: Just-in-time, 5S, Lean Engineering,…
TLS	- TOC+Lean+Six Sigma

Найбільшим досягнення синергії цих методів є подолання великих часових витрат на «переробку процесів». Кожна з них оглядає систему з трьох абсолютно різних боків і на їх межах швидко знаходяться проблеми та рішення.

РОЗДІЛ 3 ОПИС РОЗРОБЛЕНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Мета програмного забезпечення

Основною метою роботи є поліпшення ефекту батога у класичних MRP-системах. Раніше виявлено, що усі системи «виштовхуючого» керування ланцюгом поставок страждають від цього ефекту. Тож було б доцільно для подолання цього недоліку використовувати методи «витягуючих» систем.

Для збільшення гнучкості систем доцільно використати інструмент «Буфер» з теорії обмежень систем. Також основною метою є мінімізація ресурсів на складі, якої можливо домогтися за допомогою принципів ощадливого виробництва. Страховий запас виробництва є точкою, що об’єднує ці три побажання:

- Надання гнучкості системі MRP
- Використання буферів з TOC
- Мінімізація запасів LEAN

3.2 Схема роботи програми

Розроблено програму, в якій підібрано динамічний розрахунок страхового запасу для окремого ресурсу підприємства. Страховий запас ділиться на три однакові зони буферу (рисунок 3.1), для автоматизованого вибору сценарію закупівлі матеріалів.

Рисунок 3.1 Буфери страхового запасу.

Кожна зона сягає 33% від усього страхового запасу. Система миттєво реагує на зменшення запасу, що дозволяє зменшити постійні запаси матеріалів на складах. При не стандартних ситуаціях: сезонної активності, акціях, непередбачуваних обставинах, - система гнучко підлаштовується до середовища, підіймає коефіцієнт страхового запасу для кожного ресурсу індивідуально. Розглянемо схему роботи програми на рисунках 3.2 та 3.3:

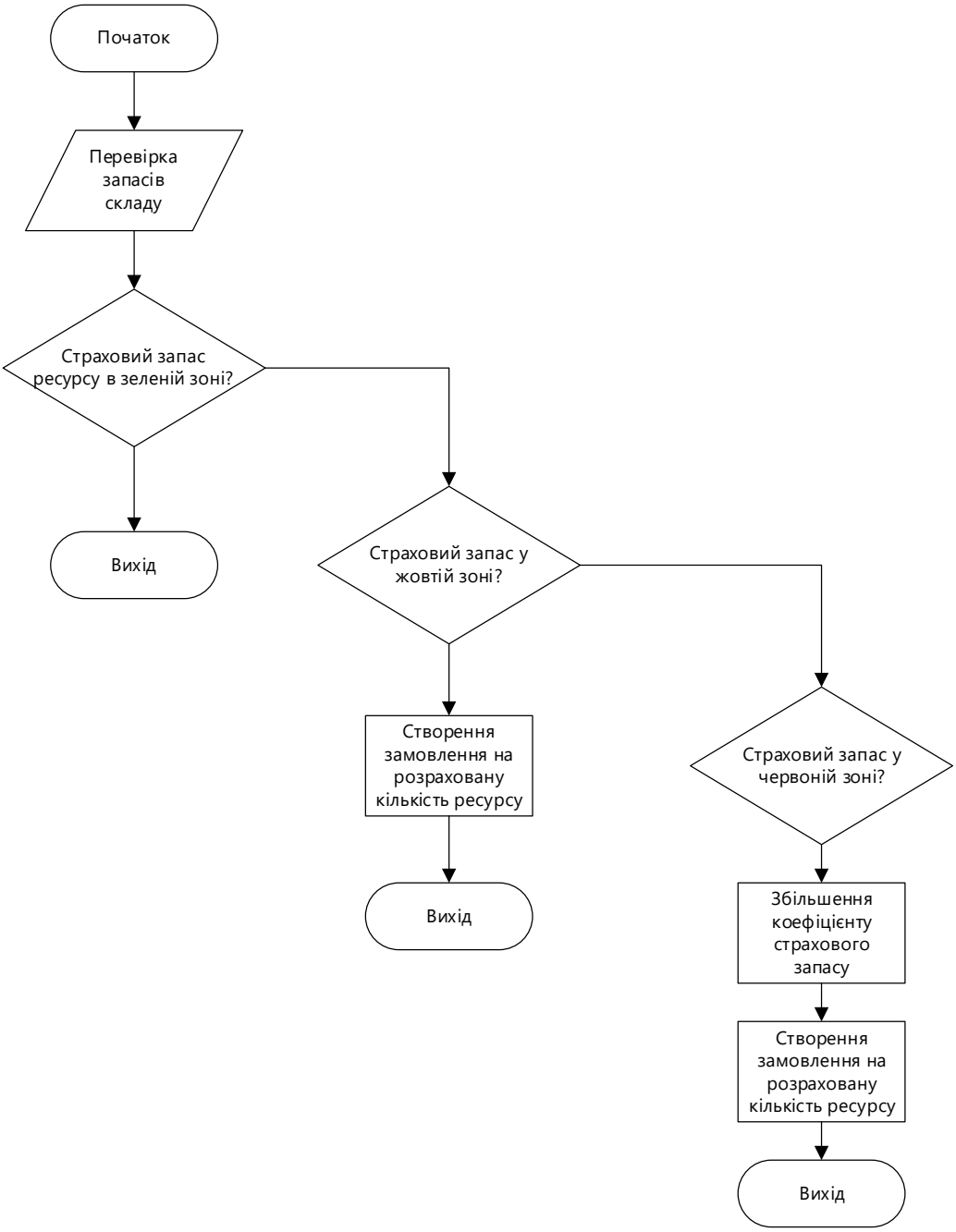


Рисунок 3.3 Схема розрахування страхового запасу та автоматичного формування замовлення.

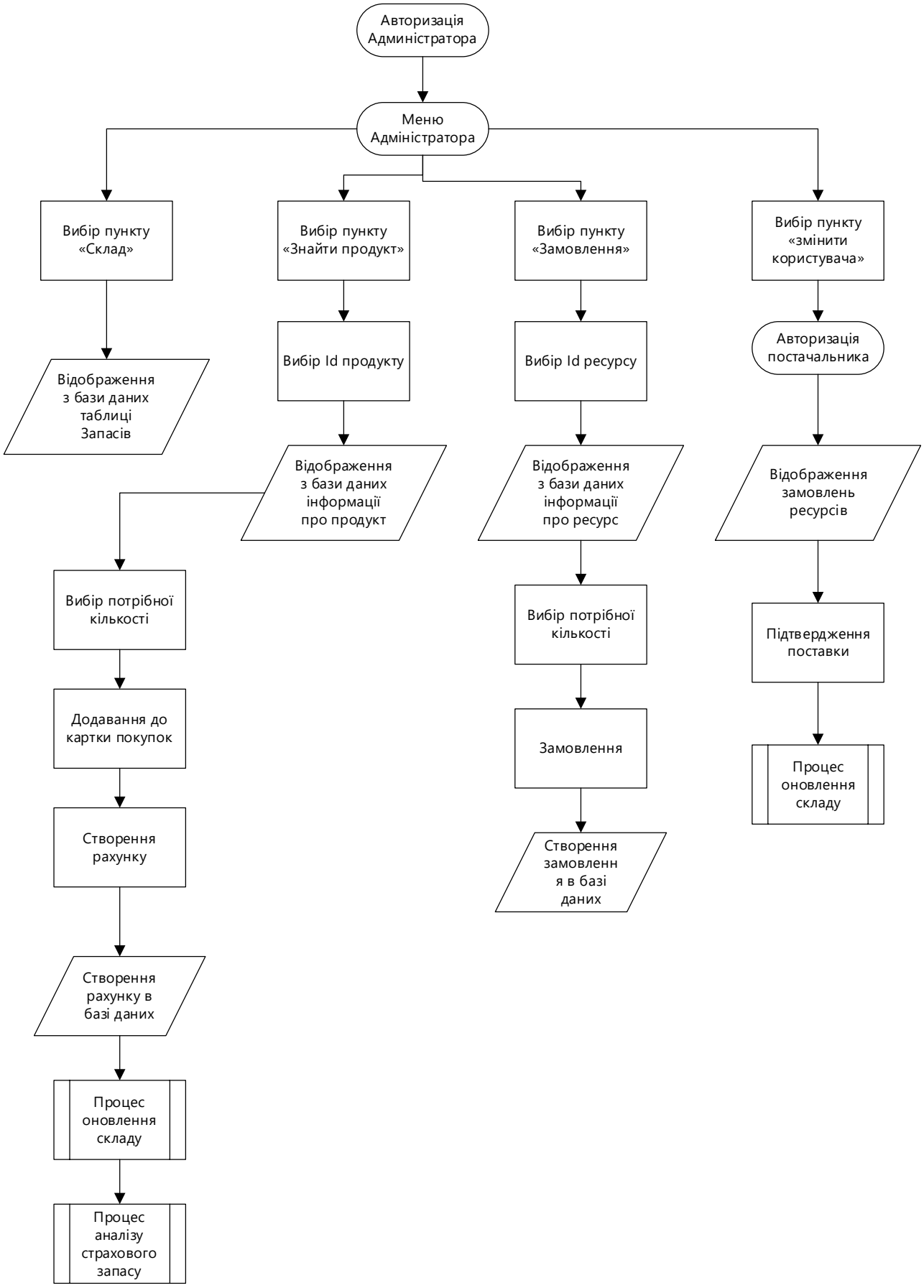


Рисунок 3.2 Схема роботи програми.

Для кожного ресурсу розраховується страховий запас S_i :

$$S_i = \sum_{j=1}^{n} a_{ij} \cdot P_{avg} \cdot t \cdot S^p \quad (3.1)$$

де:

P_{avg} – середній попит і-того продукту.

a_{ij} – елемент матриці специфікації і-того ресурсу j-того продукту.

t – середній час виконання замовлення.

S^p – коефіцієнт страхового запасу.

Коефіцієнт страхового запасу для кожного ресурсу визначається індивідуально, з врахування його особливостей та характеристик. У дослідах взято 0.5 за звичайних умов, і автоматично підвищено до 0.7 за умов різких перепадів попиту.

3.3 Відображення роботи програми

Роздивимось користувацький інтерфейс та роботу програми у різних випадках. Для початку авторизуємось у якості адміністратора (рисунок 3.4).

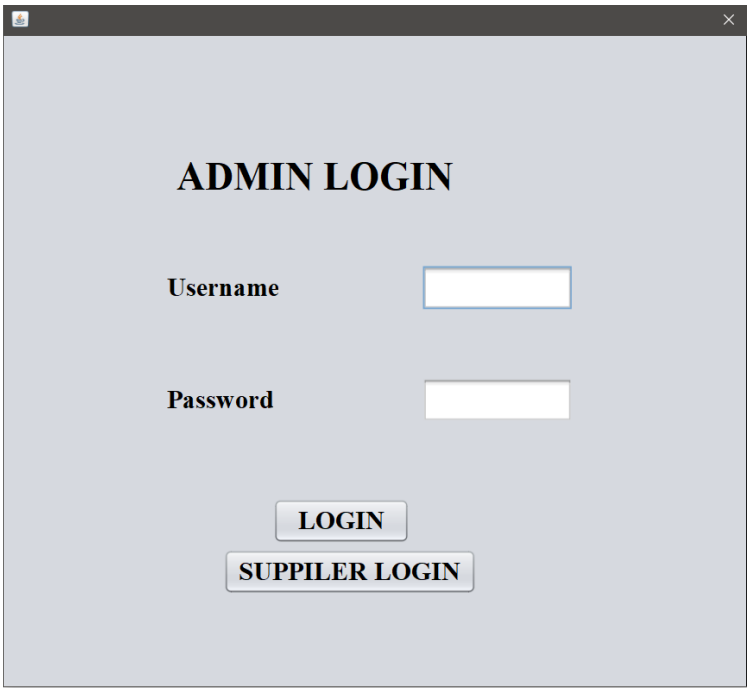


Рисунок 3.4 Сторінка авторизації адміністратора

Оберемо “Inventory” для відображення наявних ресурсів на складі (рис.3.5).

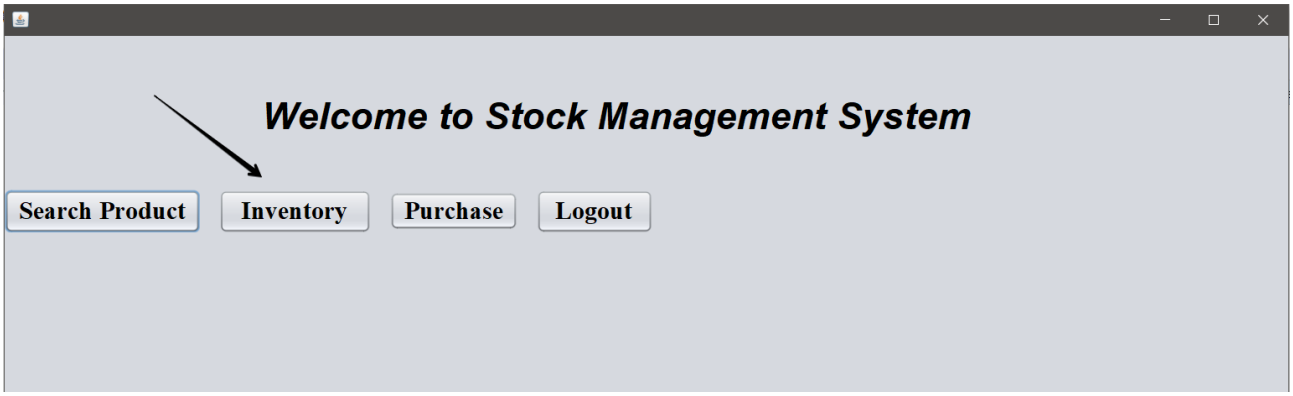


Рисунок 3.5 Сторінка користувацького меню.

Натиснемо «Показати залишки», як на рисунку 3.6.

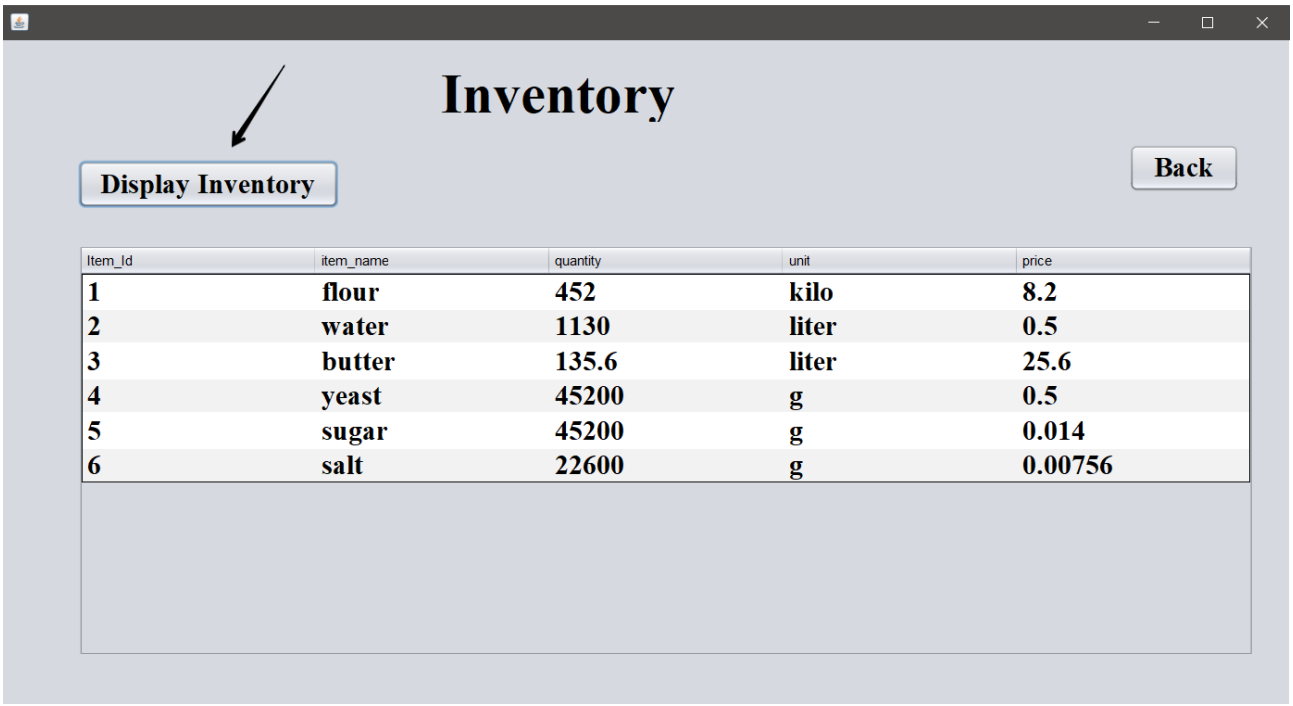


Рисунок 3.6 Сторінка «Склад», що відображає запаси на складі.

Замовимо продукт. Для цього повернемося в меню, та виберемо іншу опцію, як на рисунку 3.7.

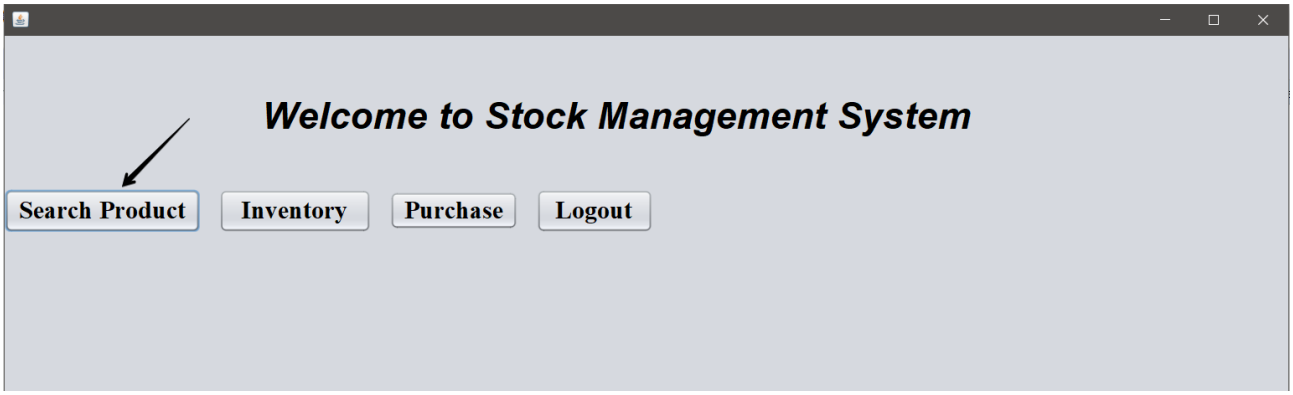


Рисунок 3.7. Сторінка користувацького інтерфейсу.

Знайдемо по I_d продукту потрібний, його дані відобразяться у вікні нижче, як на рисунку 3.8.

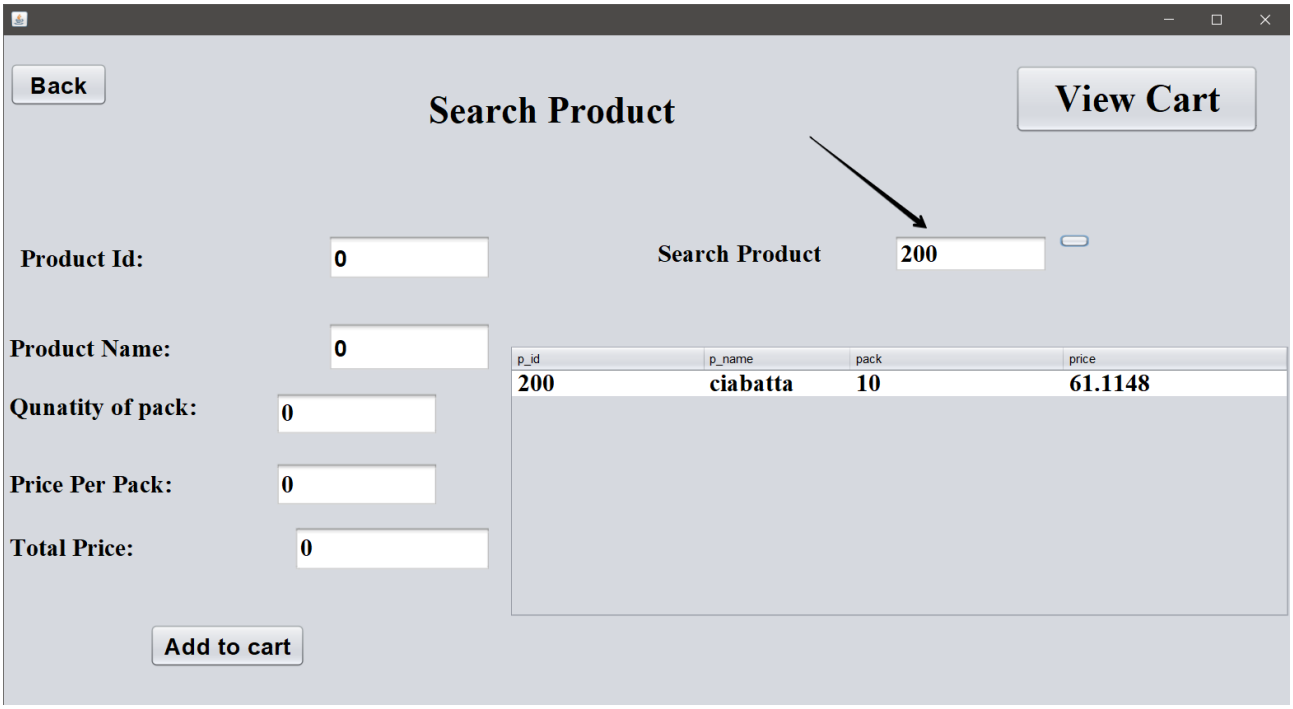


Рисунок 3.8. Сторінка «Знайти продукт».

Виберемо потрібну кількість паків. При цьому автоматично заповнюються поля, та розраховується сума замовлення, як показано на рисунку 3.9. Натиснемо “Add to cart”. Отримаємо повідомлення про успішне додавання продукту у замовлення, як на рисунку 3.10.

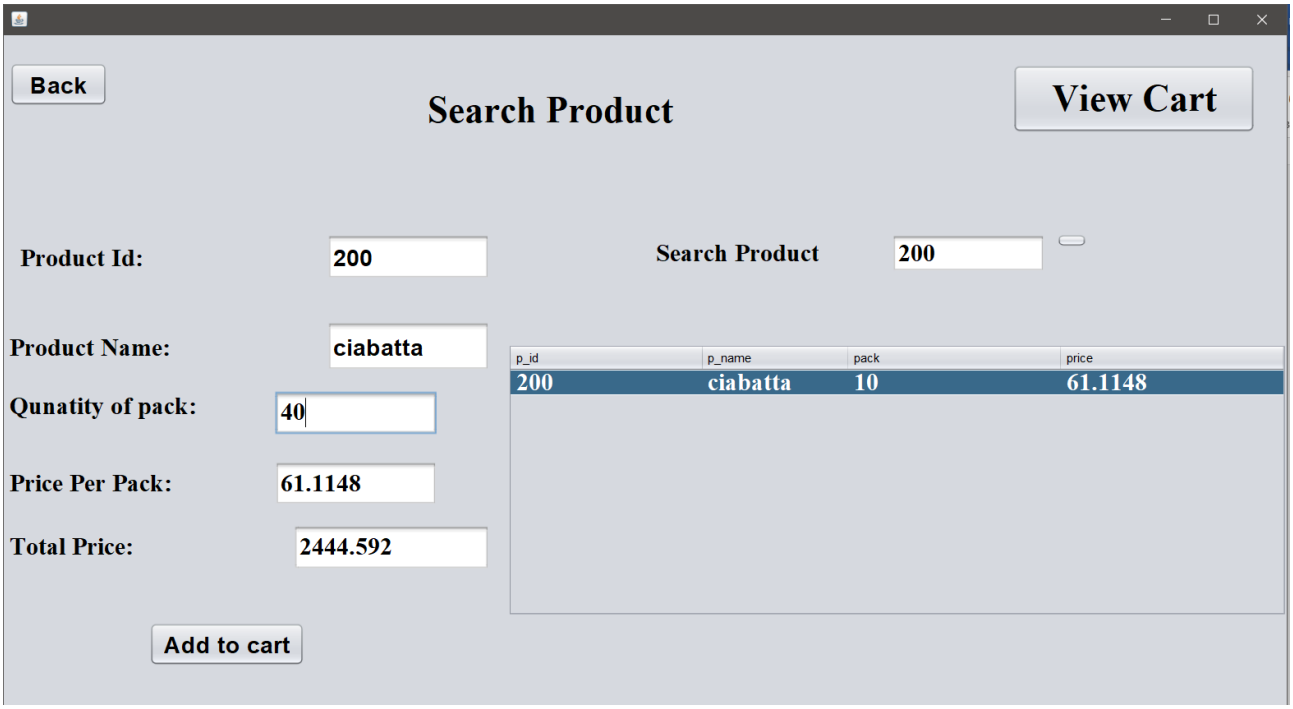


Рисунок 3.9. Відображення заповнення полів для обраного продукту.

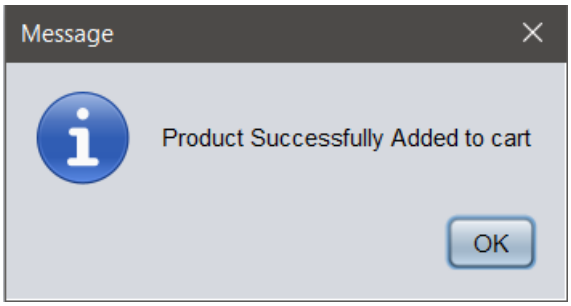


Рисунок 3.10. Повідомлення про успішне додавання продукту до картки.

Перейдемо до картки замовлення, натискаючи «показати картку», як на рисунку 3.11.

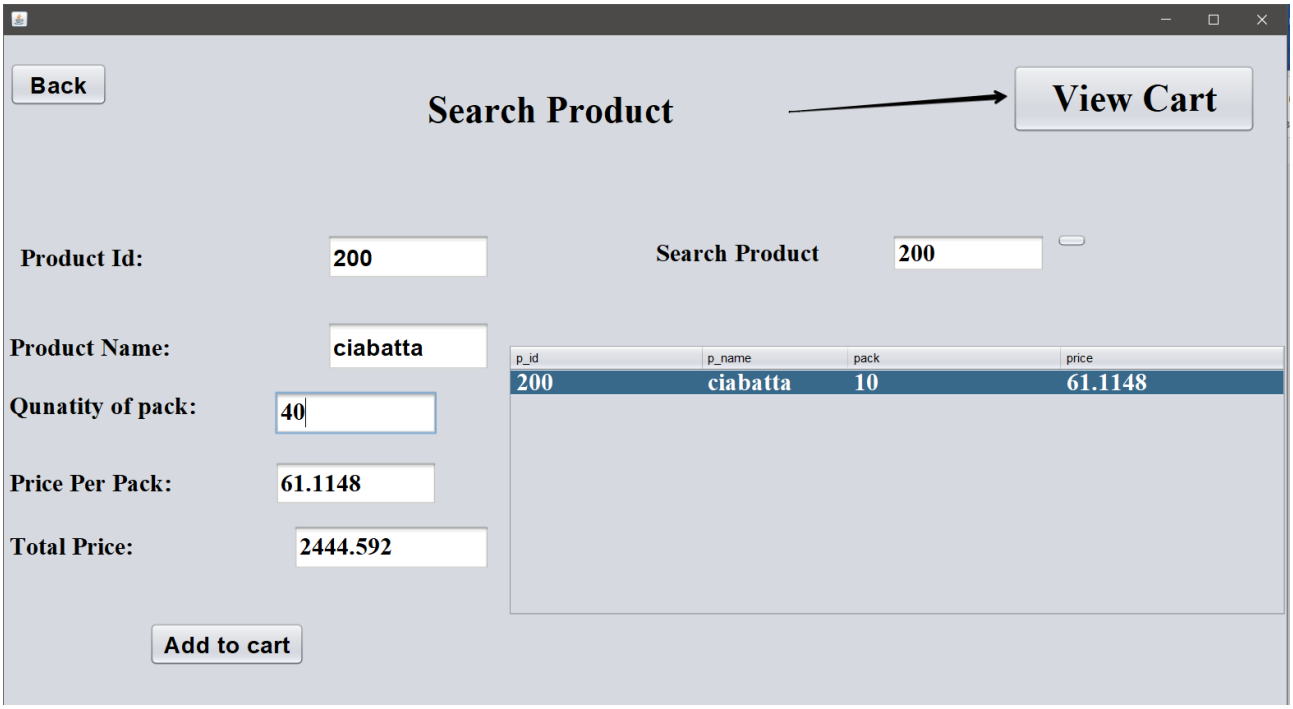


Рисунок 3.11 Перехід до сторінки «Картка».

Перейдемо до рахунку (рисунок 3.12). Тоді програма згенерує рахунок, як на рисунку 3.13.

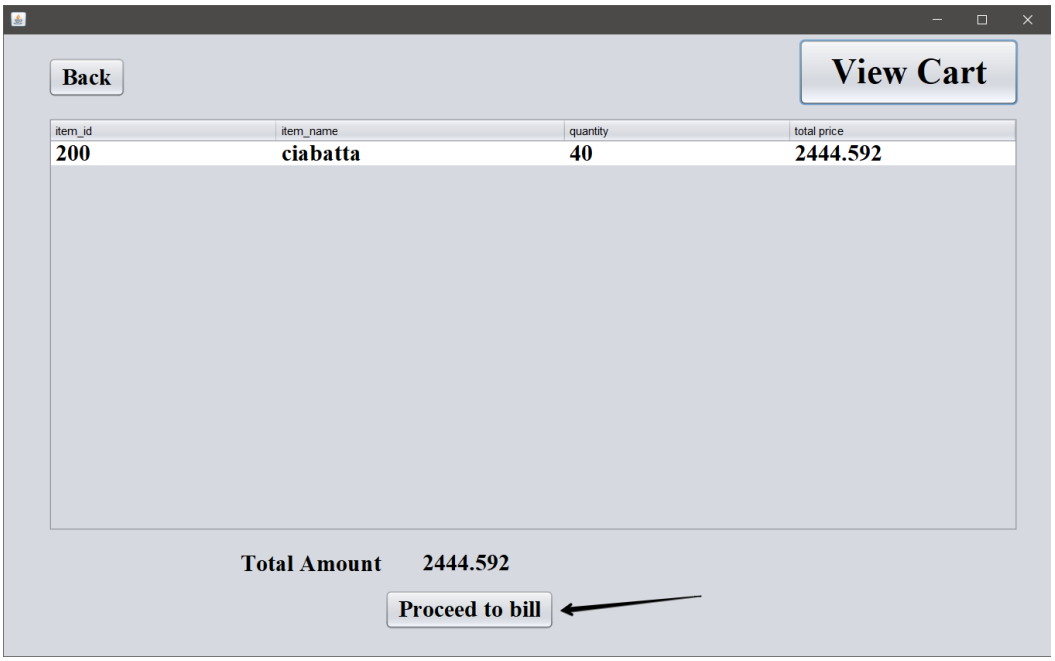


Рисунок 3.12 Сторінка «Картка».

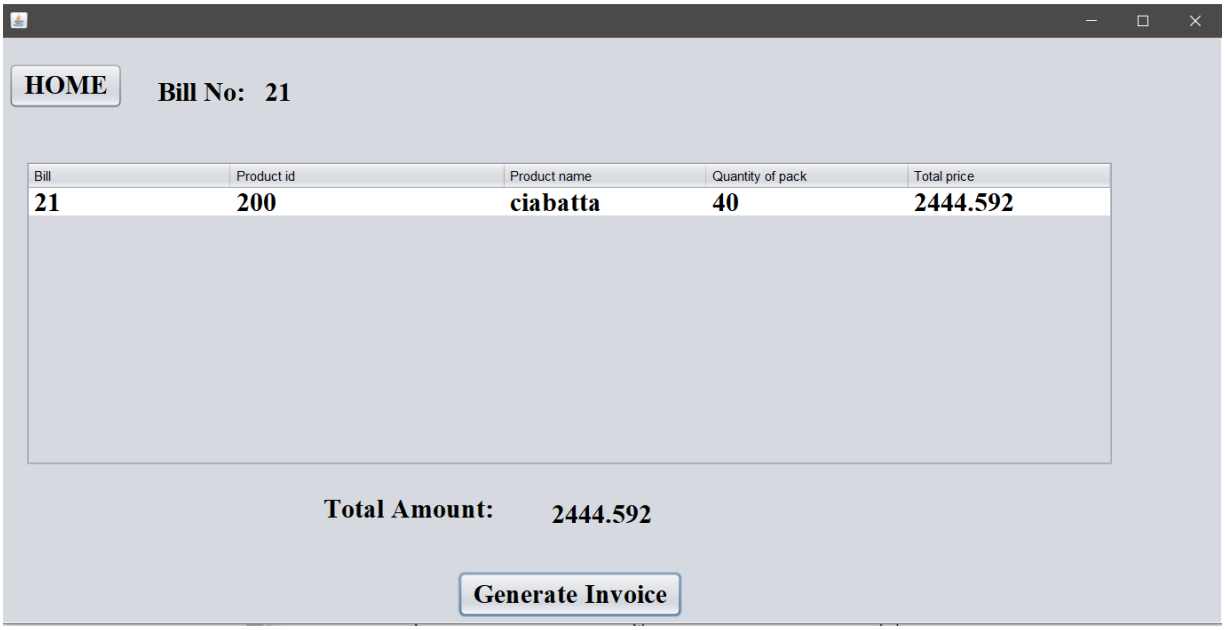


Рисунок 3.13. Сторінка «Рахунку»

Перевіримо склад. Як бачимо на рисунку 3.14 кількість ресурсів зменшилась.

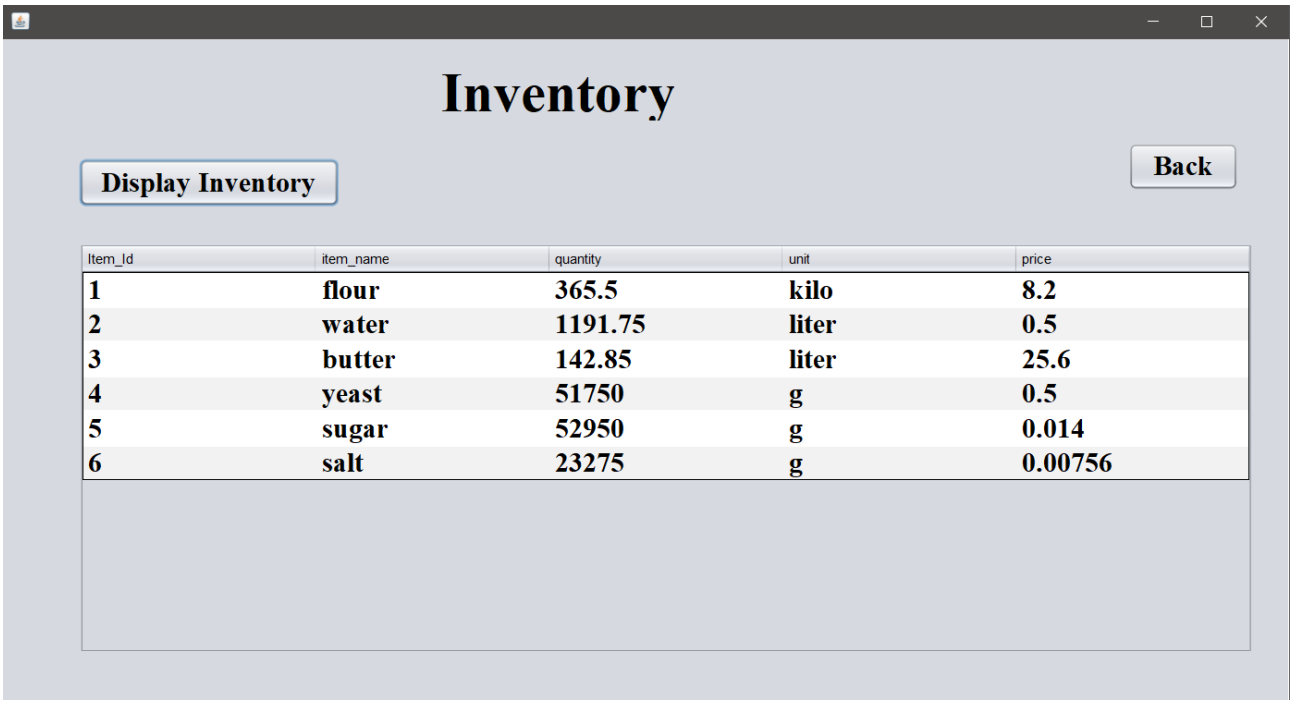


Рисунок 3.14. Сторінка «Склад», на якій зображено залишки ресурсів після замовлення.

Зробимо замовлення ресурсу. Для цього знов перейдемо до головного меню, та виберемо потрібну опцію (рисунок 3.15).

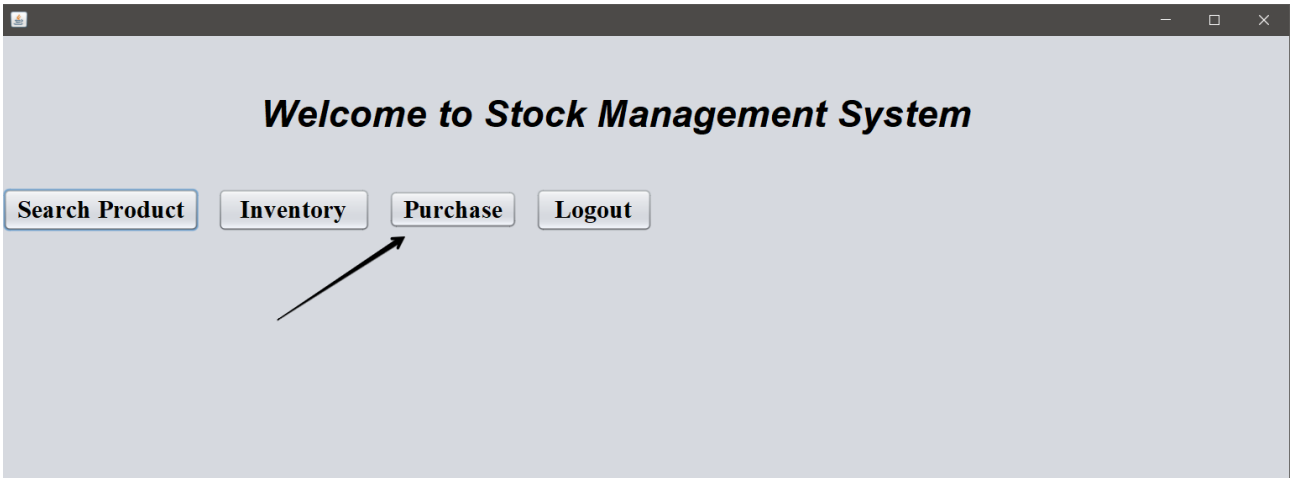


Рисунок 3.15. Сторінка користувацького меню.

Для замовлення ресурсів, необхідно також спочатку знайти на складі потрібний тип (рисунок 3.16). У вікні справа відображається його наявна кількість та ціна. Після вибору ресурсу, поля зліва заповнюються автоматично, залишається тільки обрати необхідну кількість замовлення. Ми обираємо 1 та замовляємо. Повідомлення про успішне замовлення, як на рисунку 3.17.

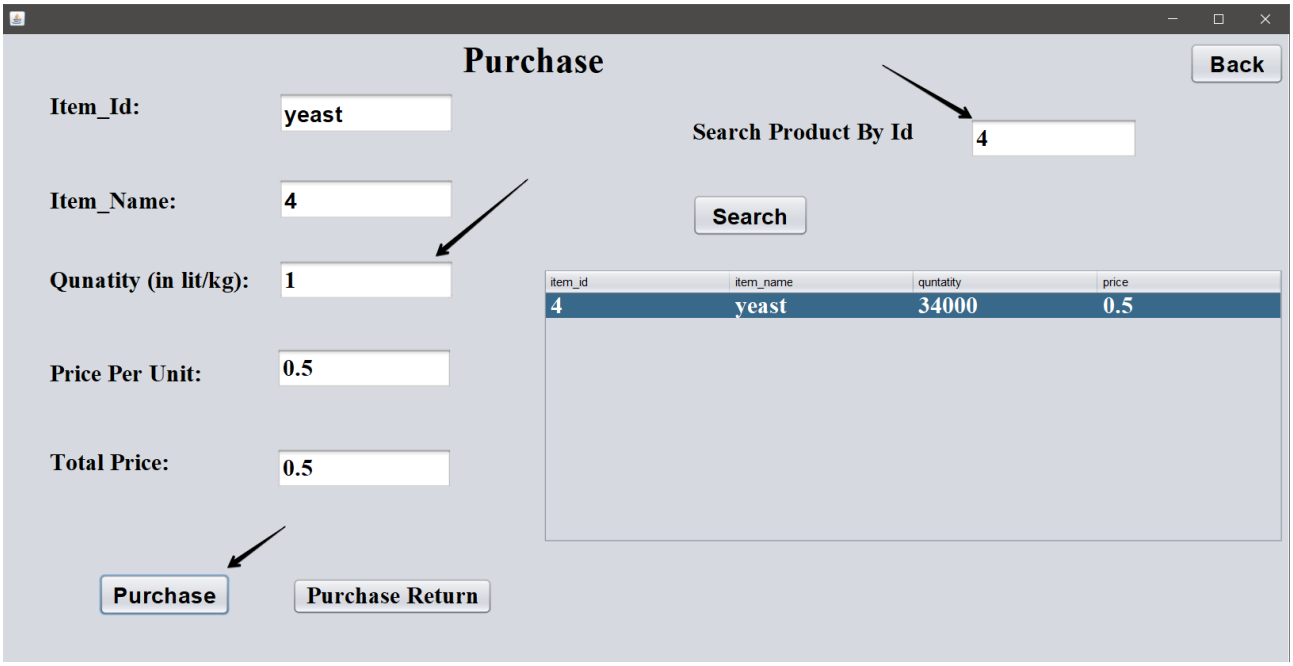


Рисунок 3.16. Сторінка замовлення ресурсів.

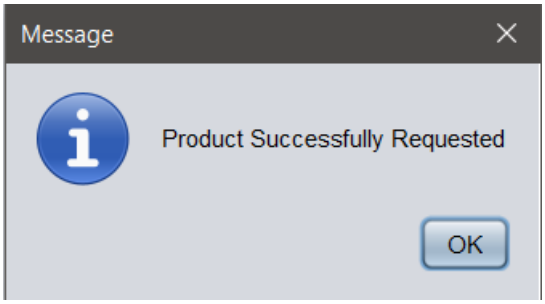


Рисунок 3.17 Повідомлення про успішне замовлення.

Подивимось на замовлення, котрі сформувались автоматично, та наше «ручне» замовлення. Для цього зайдемо в систему під логіном постачальника. Виконаємо дії як на рисунках 3.18 - 3.20.

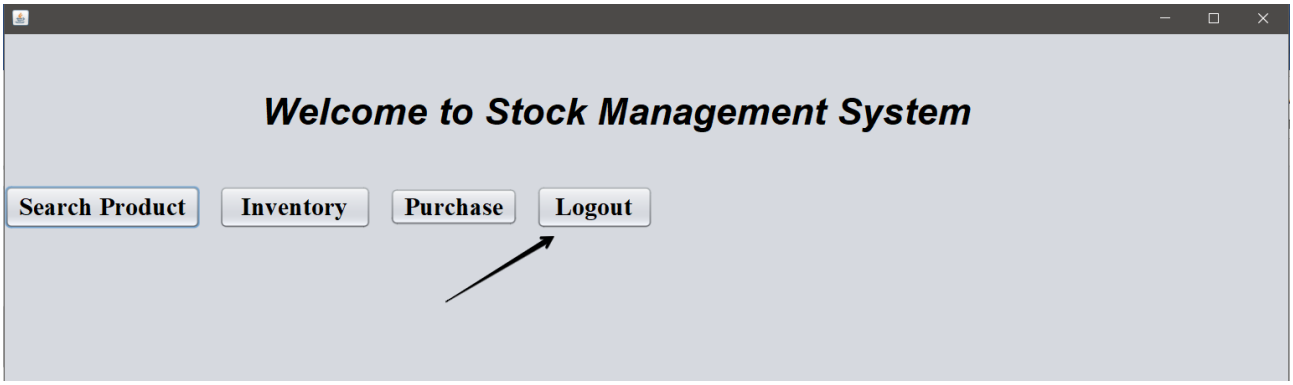


Рисунок 3.18. Сторінка користувацького меню.

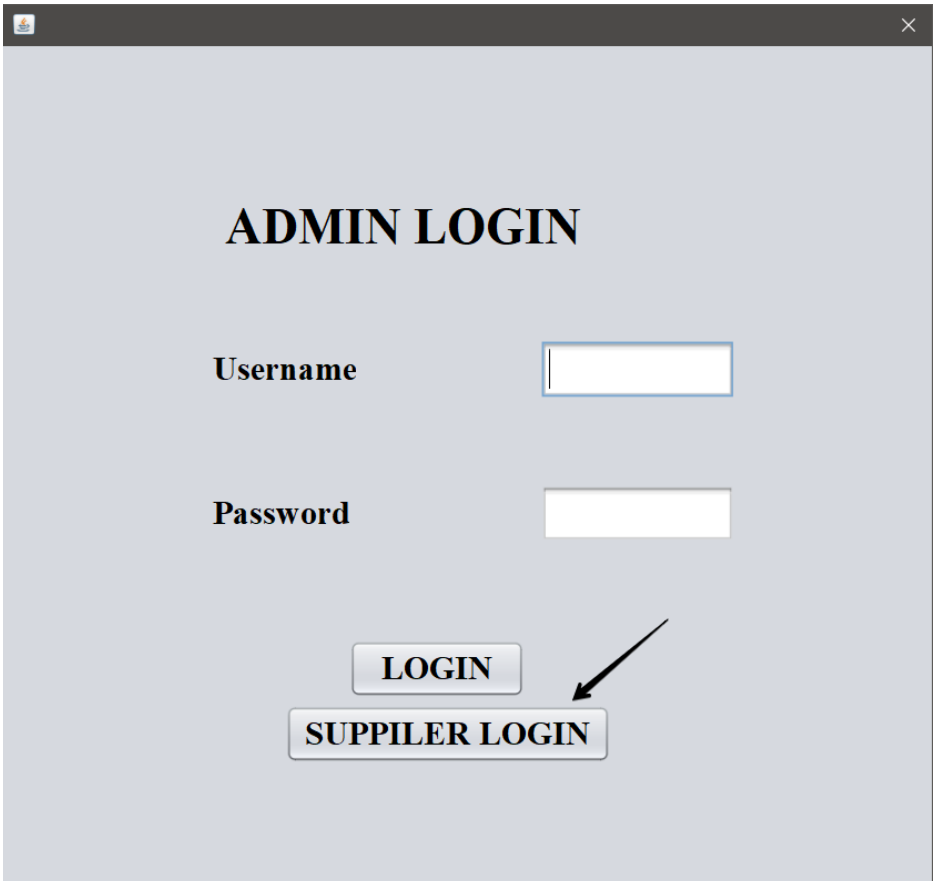


Рисунок 3.19. Сторінка авторизації користувача.

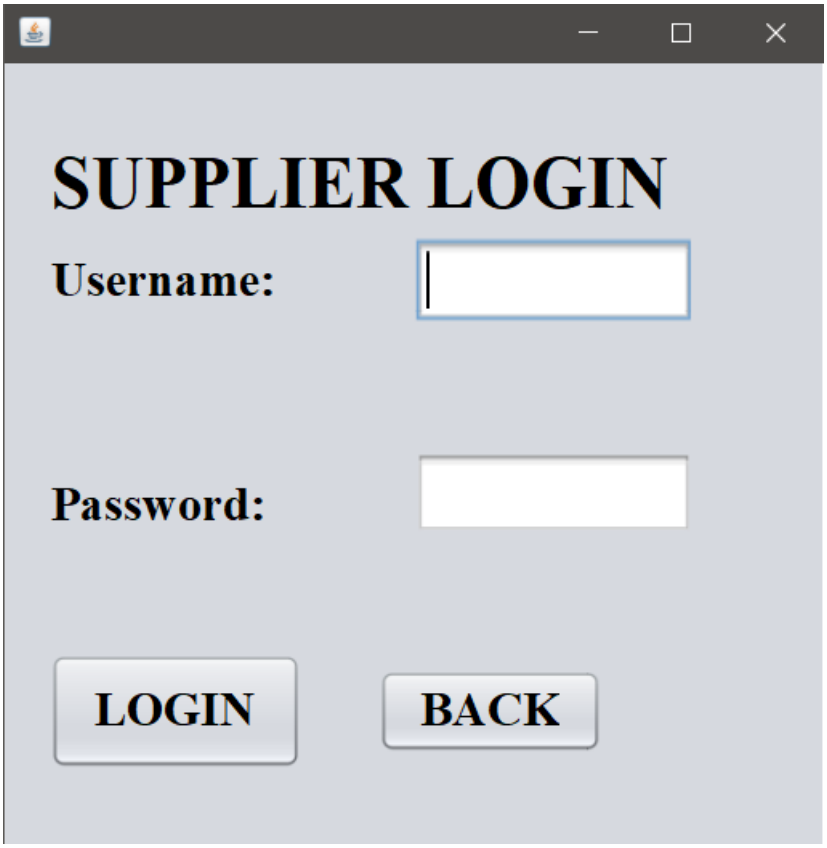


Рисунок 3.20. Сторінка авторизації постачальника.

Далі на рисунку 3.21 ми бачимо автоматичні замовлення зроблені системою (creator – auto) для відновлення усіх ресурсів до зеленої зони буферу, та наше «ручне замовлення». Унизу підраховується загальна сума поставки.

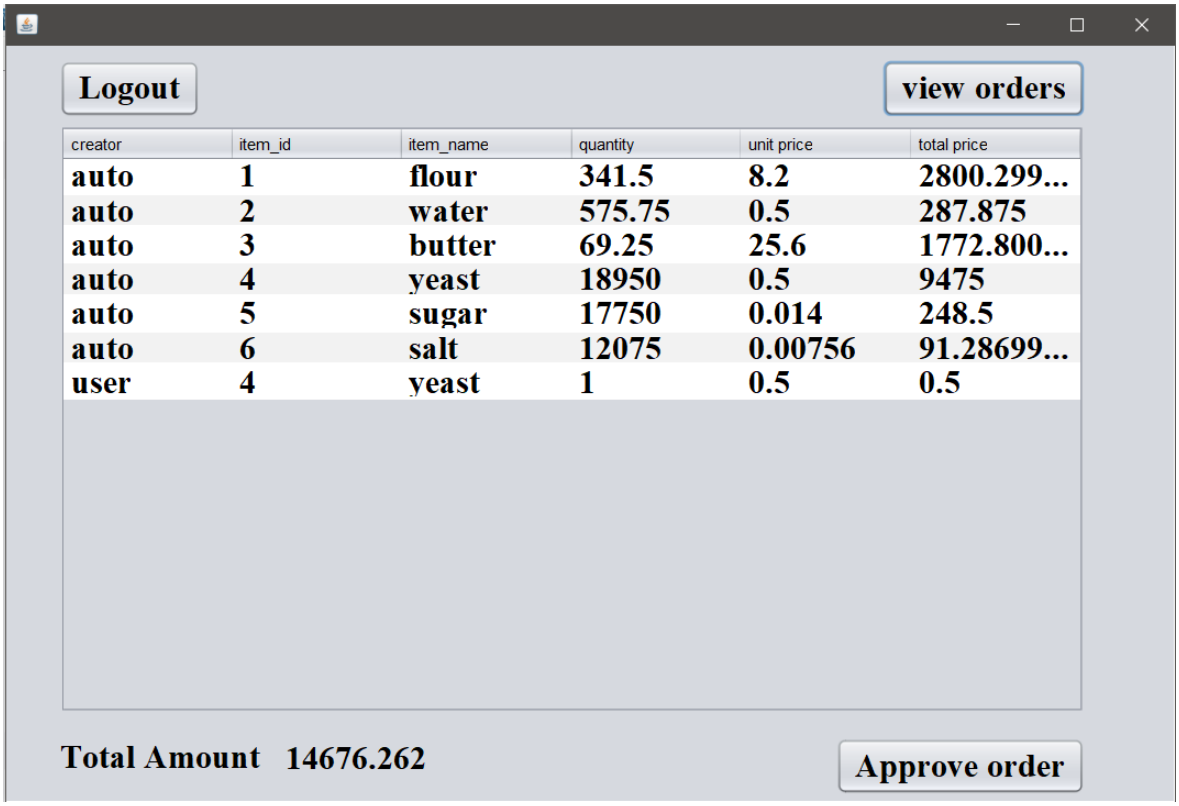


Рисунок 3.21. Сторінка замовлень.

Проведемо поставку, та перевіримо чи відновився склад. На рисунку 3.22 бачимо, що усі ресурси досягли зеленої зони буферу.

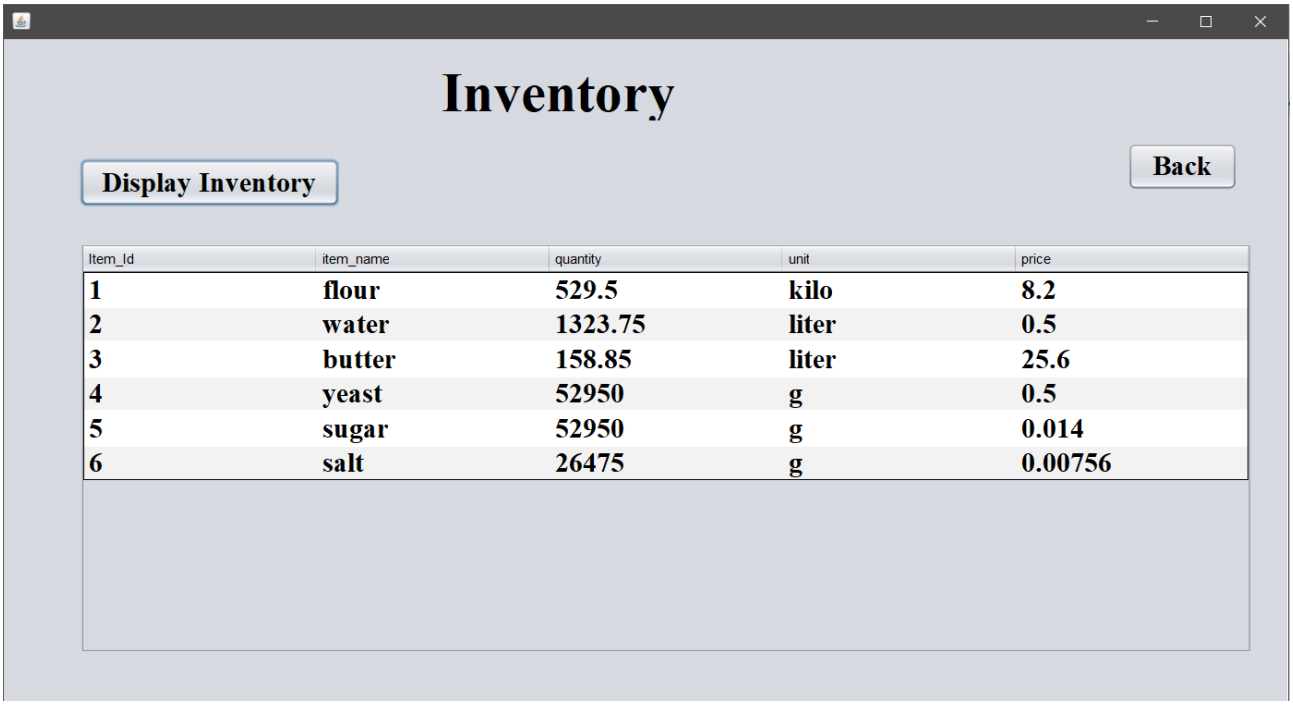


Рисунок 3.22. Сторінка «Склад».

Проведемо дослідження на даних, що відтворювали ефект батога на класичній MRP-системі. Збільшимо коливання середніх продажів на 25%, подивимось як зреагує на це система на графіку 3.23.

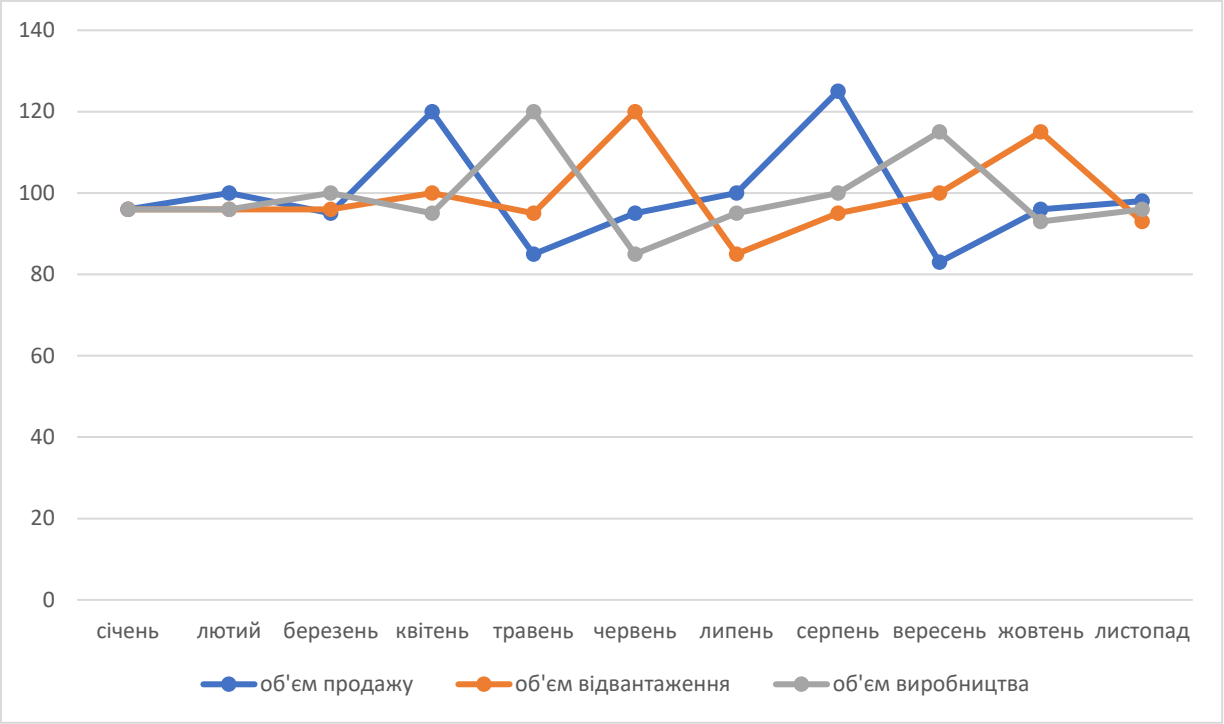


Рисунок 3.23. Графік коливання основних показників при коливанні середніх продажів на 25%.

Як бачимо, система прийшла до норми та «загасила» коливання. Збільшимо коливання середніх продажів до 30% (рисунок 3.24).

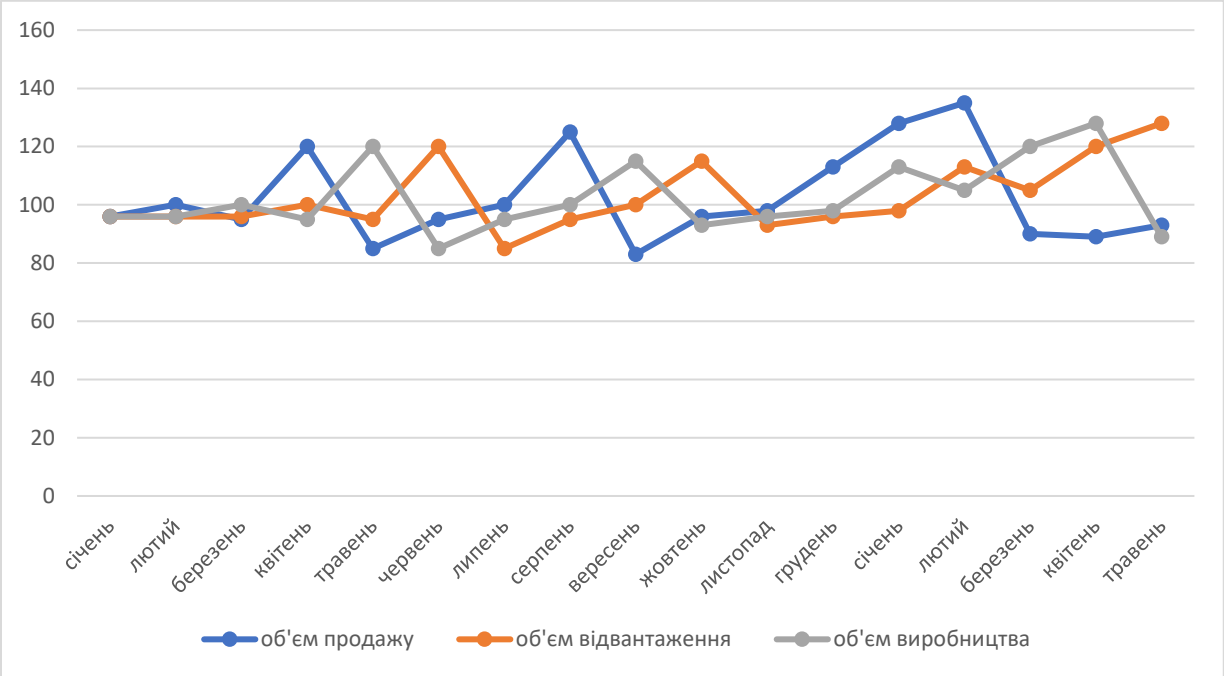


Рисунок 3.24. Графік коливання основних показників при коливанні середніх продажів на 30%.

Система відновилась, але через більший час. Отже можемо зробити висновок, що покращений метод розрахування запасу поліпшує ефект батога, але не долає його повністю.

3.4 Висновки

Розроблено програму, що поєднує у собі класичну систему MRP, інструмент Теорії обмежень систем та принципи ощадливого виробництва Lean. За допомогою цією синергії вдалося поліпшити наслідки ефекту батога, але не вдалося повністю позбавити систему від коливань.

РОЗДІЛ 4 ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ

4.1 Постановка завдання проектування

Проводиться оцінка основних параметрів, що впливають на написання скрипту програми, що імітує блок управління запасами MRP-системи з додатковими методами.

4.2. Обґрунтування функцій та параметрів програмного продукту

Цільова функція F_0 - розробка програмного продукту, що реалізує блок управління запасами на виробництві, а саме розраховує страховий запас ресурсу та постійно оновлює кількість матеріалів на складі згідно цього страхового запасу. Виходячи з конкретної задачі, можна виокремити наступні функції програмного продукту:

F1 – вибір мови програмування:

а) мова програмування Java;

б) мова програмування Python;

F2 – Вибір мови запитів до бази даних;

а) мова запитів MySQL;

б) мова запитів MSSQL;

F3 – реалізація замовлення:

а) вручну;

б) автоматично;

F4 – інтерфейс користувача:

а) Розроблений за допомогою бібліотеки Swing;

б) Веб-інтерфейс;

Варіанти реалізації основних функцій наведено у морфологічній карті (Рис. 4.1)

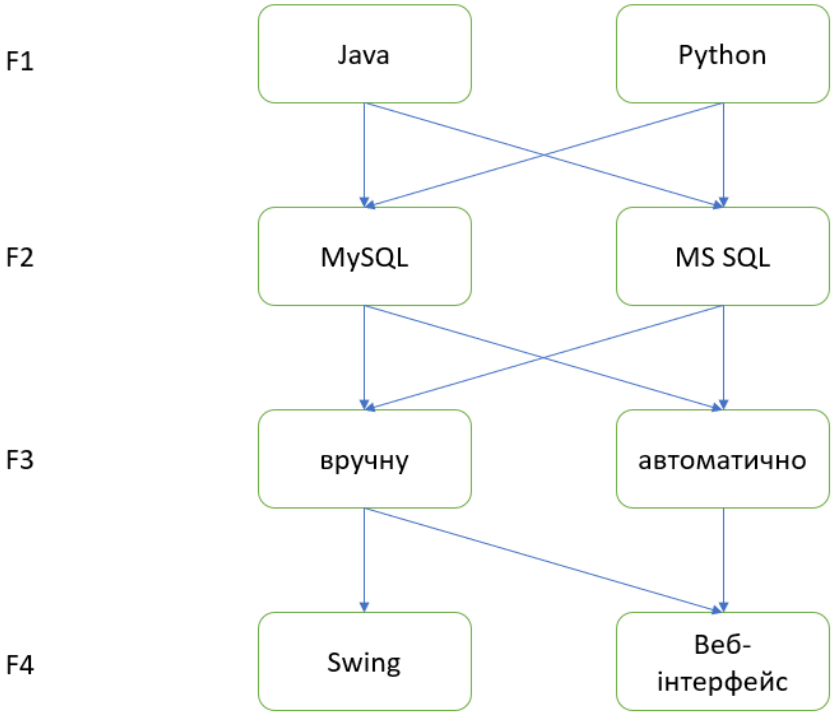


Рис 4.1 Морфологічна картка

Можливі комбінації варіантів роботи функцій (їх реалізації). Дані функції складають у собі повну множину варіантів програмного продукту.

Використовуючи дану карту, було отримано матрицю варіантів основних функцій(Таблиця 4.2).

Таблиця 4.2. Позитивно-негативна карта.

Основні функції	Варіанти реалізації	Переваги	Недоліки
F1	а	Наявність інтерфейсу, простота використання	Складність реалізації
	б	Простота реалізації	Недостатня прозорість методів
F2	а	Поширеність, універсальність	Недостатня захищеність
	б	Використовується при великому обсязі інформації	Складний у використанні
F3	а	Людський інтелектуальний аналіз даних. Гнучкість	Можливі помилки, велика втрата часу на

		системи	розрахунки та створення замовлень
	б	Точність та швидкість обчислень.	Порівняно невелика гнучкість, складність реалізації.
F4	а	Простота використання, швидкість роботи	Великий об’єм коду
	б	Естетичний вигляд, сучасність, доступність на будь-яких пристроях	Потреба у інтернет зв’язку, складність реалізації

На основі аналізу приведеної вище матриці, робимо висновок що деякі варіанти роботи були не відповідними до поставленої задачі та мають свої мінуси, а отже відповідні реалізації функцій необхідно відкинути, щоб зосередитися на правильних варіантах. Таким чином будемо використовувати наступні варіанти реалізації програмного продукту:

1. F1а – F2а – F3а – F4а;
2. F1а – F2б – F3а – F4а;
3. F1а – F2а – F3б – F4а;
4. F1а – F2б – F3б – F4а.

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

4.3 Обґрунтування системи параметрів ПП

Будемо розглядати три типи варіантів значення параметрів. Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП як показано у таблиці 4.3.

Таблиця 4.3. Основні параметри програмного продукту.

Назва	Умовні	Одиниці	Значення параметра

параметра	позначення	виміру			
			Гірші	Середні	Кращі
Час розробки алгоритму	X1	Год	85	55	35
Час відповіді бази даних на запити	X2	Мс	1000	850	400
Об’єм пам’яті для збереження даних	X3	Мб	16	8	2
Потенційний об’єм програмного коду	X4	кількість комірок коду	4000	3000	2500

За даними таблиці 4.3 будуються графічні характеристики параметрів – рис. 4.4 – рис. 4.7.

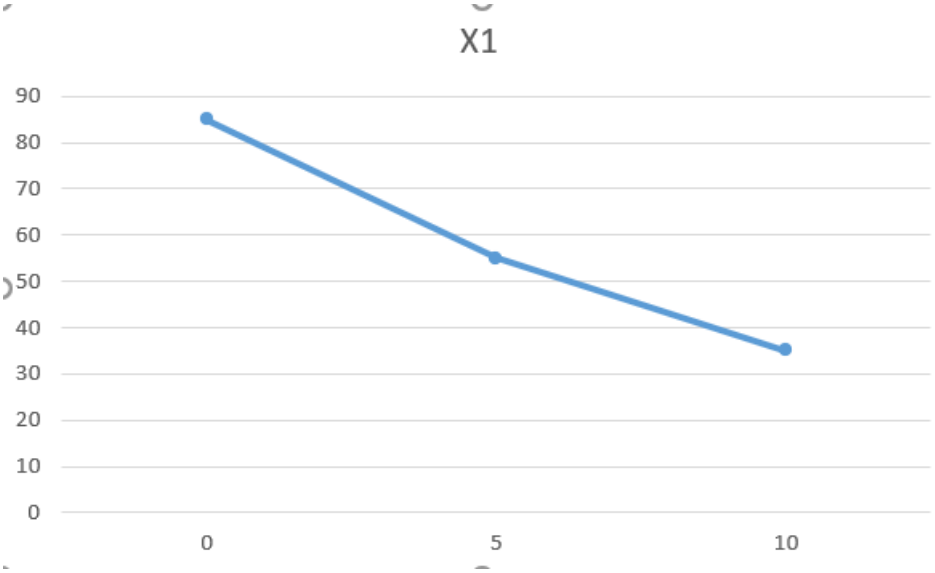


Рис. 4.4. X1, час розробки алгоритму.

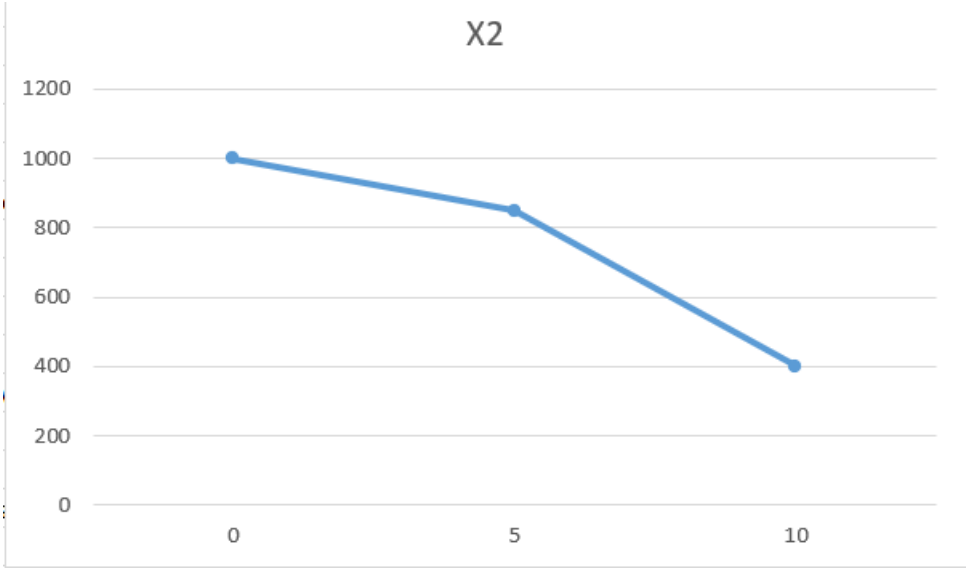


Рис. 4.5. X2, час відповіді бази даних на запити.

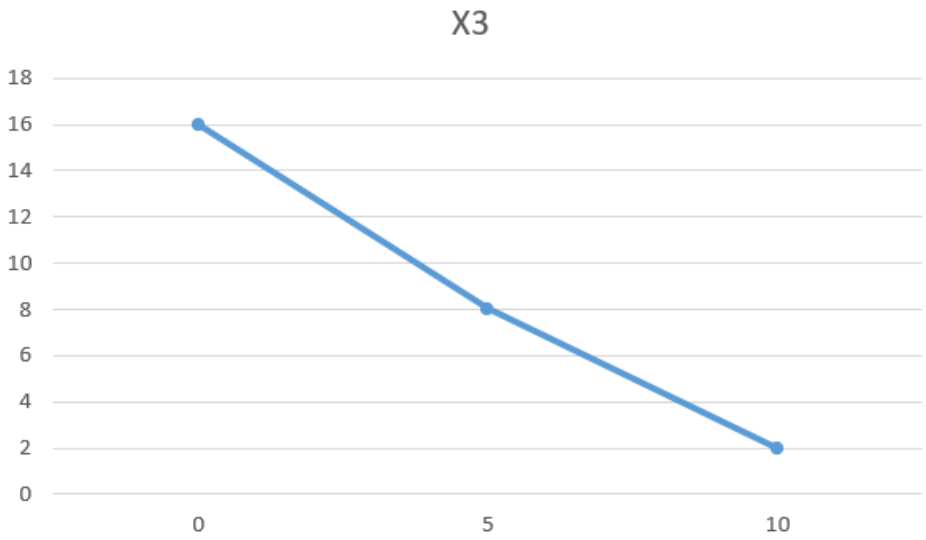


Рис. 4.6. X3, об’єм пам’яті для збереження даних.

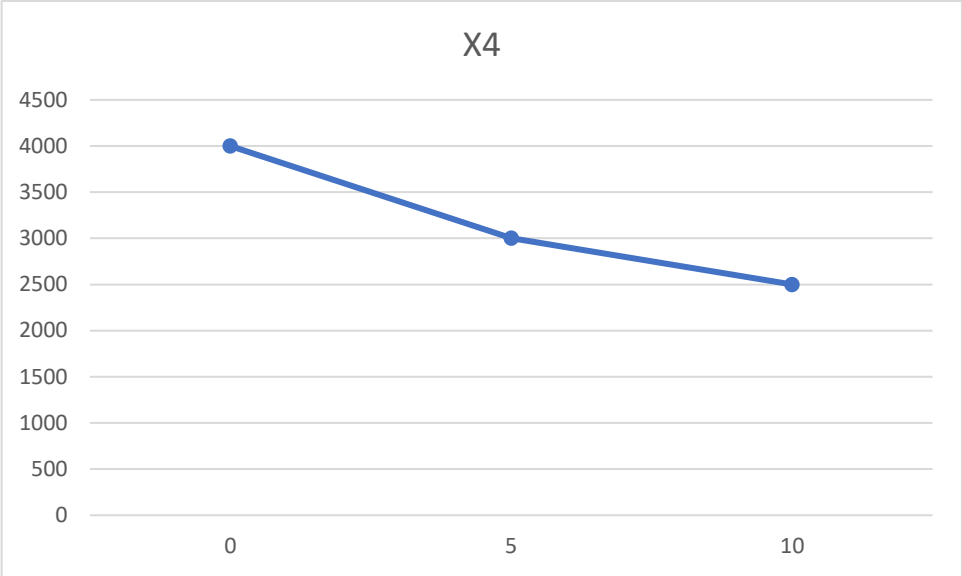


Рис. 4.7. X4, потенційний об’єм програмного коду.

Результати експертного ранжування наведені у таблиці 4.8.

Таблиця 4.8. Результати ранжування параметрів.

Познач. Параметра	Ранг параметра за оцінкою експерта							Сума рангів R_i	Відхилення Δ_i	Δ_i^2
	1	2	3	4	5	6	7			
X1	2	1	2	1	1	1	1	9	-8,5	72,25
X2	4	4	3	4	4	4	4	27	9,5	90,25
X3	3	3	4	3	3	3	3	22	4,5	20,25
X4	1	2	1	2	2	2	2	12	-5,5	30,25
Разом	10	10	10	10	10	10	10	70	0	213

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3-n)}, W = \frac{12*213}{(7^2*(4^3-4))} = 0{,}87 > W_k = 0{,}67. \quad (4.5)$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 4.9. За найбільший ранг приймаємо 4.

Таблиця 4.9. Попарне порівняння показників. - –

Параметри	**Експерти**							**Кінцева оцінка**	**Числове Значення**
	1	**2**	**3**	**4**	**5**	**6**	**7**		
X1 та X2	<	<	<	<	<	<	<	<	0.5
X1 та X3	<	<	<	<	<	<	<	<	0.5
X1 та X4	>	<	>	<	<	<	<	<	0.5
X2 та X3	>	>	<	>	>	>	>	>	1.5
X2 та X4	<	<	<	<	<	<	<	<	0.5
X3 та X4	>	>	>	>	>	>	>	>	1.5

Таблиця 4.10. Розрахунок вагомості параметрів.

Параметри	Параметри				Перша ітер.		Друга ітер.		Третя ітер	
	X1	X2	X3	X4	b_i	$К_{вi}$	b_i^1	$К_{вi}^1$	b_i^2	$К_{вi}^2$
X1	1,0	0,5	0,5	0,5	2,5	0,15625	9,25	0,151639	35,125	0,151401
X2	1,5	1,0	1,5	0,5	4,5	0,28125	17,25	0,282787	65,625	0,282866
X3	1,5	0,5	1,0	1,5	4,5	0,28125	17,25	0,282787	65,625	0,282866
X4	1,5	1,5	0,5	1,0	4,5	0,28125	17,25	0,282787	65,625	0,282866
Всього:					16	1	61,00	1	232,00	1

Як видно з таблиці, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Варіант функції F1 описується параметром X1, F2 параметром X2, F3 параметром X3, F4 параметром X4.

4.4 Аналіз рівня якості варіантів реалізації функцій

Таблиця 4.11. Розрахунок показників рівня якості варіантів реалізації.

Основні	Варіант	Абсолютне	Бальна	Коефіцієнт	Коефіцієнт

функції	реалізації функції	значення параметра	оцінка параметра	вагомості параметра	рівня якості
F1	А	65	3,3	0.151	0,5
F2	А	800	5,5	0.283	1,56
	Б	900	3,3	0.283	0,93
F3	А	2	10	0.283	2,83
	Б	3	9,1	0.283	2,5753
F4	А	3300	3,5	0.283	0,99

1. F1(а)=>F2(а)=>F3(а)=>F4(а)
2. F1(а)=>F2(б)=>F3(а)=>F4(а)
3. F1(а)=>F2(а)=>F3(б)=>F4(а)
4. F1(а)=>F2(б)=>F3(б)=>F4(а)

За даними з таблиці 4.11 визначаємо рівень якості кожного з варіантів:

$К_{я1}$ = 0,5 + 1,56 + 2,83 + 0,99 = 5,88

$К_{я2}$ = 0,5 + 0,93 + 2,83 + 0,99 = 5,25

$К_{я3}$ = 0,5 + 1,56 + 2,57 + 0,99 = 5,62

$К_{я4}$ = 0,5 + 0,93 + 2,57 + 0,99 = 4,99

Отже, найкращим є перший варіант, для якого коефіцієнт технічного рівня має найбільше значення.

Для оцінки трудомісткості розробки спочатку проведемо розрахунок трудомісткості. Усі варіанти мають наступні основні завдання:

1) Розробка проекту програмного продукту
2) Розробка програмної оболонки

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до групи Б. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1; а в завданні 2 – до групи 3.

3.1) завантаження даних із бд MySQL запитами;

3.2) завантаження даних із бд MS SQLзапитами;

4.1) Створення замовлення вручну;

4.2) Автоматичне створення замовленняцт

В варіанті 1 присутні наступні додаткові завдання під номерами 3.1 та 4.1

В варіанті 2 присутні наступні додаткові завдання під номерами 3.2 та 4.1

В варіанті 3 присутні наступні додаткові завдання під номерами 3.1 та 4.2

В варіанті 4 присутні наступні додаткові завдання під номерами 3.2 та 4.2

За ступенем новизни до групи А відноситься задання 1, 4.2, до групи Б – 4.1, до групи В відносяться завдання 3.2, 3.1 , до групи Г — 2.

За складністю алгоритмів до групи 1 відносяться завдання 1, 4.2 до групи 2 відноситься завдання 2, 4.1, до групи 3 – 3.1, 3.2.

Якщо при розробці ПП використовуються стандартні бібліотеки та(чи) пакети прикладних програм, норми часу коректуються з допомогою коефіцієнта Кст,значення якого лежать в діапазоні 0,6-0,8. У нашому випадку Кст =0,7.

Для першого завдання (степінь новизни А, група складності – 1), тобто T_P=90 людино-днів, $К_П$ = 1,7, $К_{СК}$ = 1; $К_{СТ}$ = 0.7.

Тоді, загальна трудомісткість програмування першого завдання дорівнює:

$$T_1 = 90 \cdot 1,7 \cdot 1 \cdot 0,7 = 107,1 \text{ людино-днів}$$

Для другого завдання (степінь новизни Г, група складності – 2), тобто T_P= 12 людино-днів, $К_П$ = 0,43; $К_{СК}$ = 1; $К_{СТ}$ = 0.7:

$$T_2 = 12 \cdot 0,43 \cdot 1 \cdot 0,7 = 3,61 \text{ людино-днів.}$$

Для завдання 3.1 (степінь новизни В, група складності – 3), тобто T_P=12 людино-днів, $К_П$ = 0,6 ; $К_{СК}$ = 1; $К_{СТ}$ = 0.7:

$$T_{3.1} = 12 \cdot 0,6 \cdot 1 \cdot 0.7 = 5,04 \text{ людино-днів.}$$

Для завдання 3.2(степінь новизни В, складність алгоритмів 2) , тобто T_P= 19 людино-днів, $К_П$ =0,6 ; $К_{СК}$ = 1; $К_{СТ}$ = 0.7

$$T_{3.2} = 12 \cdot 0,6 \cdot 1 \cdot 0.7 = 5,04 \text{ людино-днів.}$$

Для завдання 4.1 (степінь новизни Б, група складності – 2), тобто T_P=27 людино-днів,$К_П$ = 1.08; $К_{СК}$ = 1; $К_{СТ}$ = 0.7:

$$T_{4.1} = 27 \cdot 1,08 \cdot 1 \cdot 0.7 = 20, 41 \text{ людино-днів.}$$

Для завдання 4.2(степінь новизни А, група складності — 1), тобто T_P=90 людино-днів, $К_П$ = 1,7 , $К_{СК}$ = 1; $К_{СТ}$ = 0.7.

$$T_{4.2} = 90 \cdot 1,7 \cdot 1 \cdot 0,7 = 107,1 \text{ людино-днів}$$

Складаємо трудомісткість відповідних завдань для кожного з обраних

варіантів реалізації програми, щоб отримати їх трудомісткість:

T_I = (107,1 + 3,61 + 5,04+20,41) ·8 = 1089,28 людино-годин;

T_{II} = (107,1 + 3,61 + 5,04 + 20, 41) ·8 = 1089,28 людино-годин.

T_{III} = (107,1 + 3,61 + 5,04 + 107,1) ·8 = 1782,8 людино-годин.

T_{IV} = (107,1 + 3,61 + 5,04 + 107,1) ·8 = 1782,8 людино-годин.

Найбільш високу трудомісткість має варіант IV та III.

В розробці беруть участь два програмісти з окладом 15000 грн., один фінансовий аналітик з окладом 8000 грн. Визначимо зарплату за годину:

$$СЧ = \frac{15000 + 15000 + 8000}{3 \cdot 21 \cdot 8} = 75{,}4 \text{ грн.}$$

Тоді, заробітну плата розробників за варіантами становить:

I. СЗП = 75,4 · 1089,28 · 1.2 = 98558,05 грн.;

II. СЗП = 75,4 · 1089,28 · 1.2 = 98558,05 грн.

III. СЗП = 75,4 · 1782,8 · 1.2 = 161307,74 грн.

IV. СЗП = 75,4 · 1782,8 · 1.2 = 161307,74 грн.

Відрахування на єдиний соціальний внесок становить 22%:

I. СВІД = СЗП · 0.22 = 98558,05 · 0.22 = 21682,77 грн.

II. СВІД = СЗП · 0.22 = 98558,05 · 0.22 = 21682,77 грн.

III. СВІД = СЗП · 0.22 = 161307,74 · 0.22 = 35487,7 грн.

IV. СВІД = СЗП · 0.22 = 161307,74 · 0.22 = 35487,7 грн.

Тепер, визначимо витрати на оплату однієї машино-години. ($С_М$)

Так як одна ЕОМ обслуговує одного програміста з окладом 15000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 15000 \cdot 0{,}2 = 36000 \text{ грн.}$$

З урахуванням додаткової заробітної плати:

$$С_{ЗП} = С_Г \cdot (1 + К_З) = 36000 \cdot (1 + 0.2) = 43200 \text{ грн.}$$

Відрахування на єдиний соціальний внесок:

$$С_{ВІД} = С_{ЗП} \cdot 0.22 = 43200 \cdot 0{,}22 = 9504 \text{ грн.}$$

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 19999,9 грн.

$$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.15 \cdot 0.25 \cdot 19999,9 = 5749,97 \text{ грн.},$$

Витрати на ремонт та профілактику розраховуємо як:

$$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.15 \cdot 19999,9 \cdot 0.05 = 1149,99 \text{ грн.},$$

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 142 – 16) \cdot 8 \cdot 0.9 = 1324,8 \text{ год.},$$

Витрати на оплату електроенергії розраховуємо за формулою:

$$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН} = 1324,8 \cdot 0,156 \cdot 0,61 \cdot 1,75 = 220,62 \text{ грн.},$$

Накладні витрати розраховуємо за формулою:

$$С_Н = Ц_{ПР} \cdot 0.67 = 19999,9 \cdot 0,67 = 13399,9 \text{ грн.}$$

Тоді, річні експлуатаційні витрати будуть:

$$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н$$

$$С_{ЕКС} = 43200 + 9504 + 5749,97 + 1149,99 + 220,62 + 13399,9 = 73224,55 \text{ грн.}$$

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$$С_{М\text{-}Г} = С_{ЕКС} / Т_{ЕФ} = 73224,55 / 1324,8 = 55,27 \text{ грн/час.}$$

Отже, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$$С_М = С_{М\text{-}Г} \cdot Т$$

I. $С_М = 55,27 \cdot 1089,28 = 60204,5$ грн.

II. $С_М = 55,27 \cdot 1089,28 = 60204,5$ грн.

III. $С_М = 55,27 \cdot 1782,8 = 98535,36$ грн.

IV. $С_М = 55,27 \cdot 1782,8 = 98535,36$ грн.

Накладні витрати складають 67% від заробітної плати:

$$С_Н = С_{ЗП} \cdot 0,67$$

I. $С_Н = 98558,05 \cdot 0,67 = 66033,89$ грн.

II. $С_Н = 98558,05 \cdot 0,67 = 66033,89$ грн.

III. $С_Н = 161307,74 \cdot 0,67 = 108076,19$ грн.

IV. $С_Н = 161307,74 \cdot 0,67 = 108076,19$ грн.

Отже, вартість розробки ПП за варіантами становить:

$С_{ПП} = С_{ЗП} + С_{ВІД} + С_{М} + С_{Н}$

I. $С_{ПП}$ = 98558,05 +21682,77 +60204,5 + 66033,89 = 246479,21 грн.;

II. $С_{ПП}$ = 98558,05 + 21682,77 +60204,5 + 66033,89 =246479,21 грн.

III. $С_{ПП}$ = 161307,74 + 35487,7+98535,36+108076,19 = 403406,99 грн.

IV. $С_{ПП}$ = 161307,74 + 35487,7+ 98535,36+108076,19 = 403406,99 грн.

Вибір кращого варіанта ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$К_{ТЕРj} = К_{Кj} / С_{ППj}$,

$К_{ТЕР1} = 5,88 / 246479,21 = 2,38 \cdot 10^{-5}$

$К_{ТЕР2} = 5,25 / 246479,21 = 2,13 \cdot 10^{-5}$

$К_{ТЕР3} = 5,62 / 403406,99 = 1,39 \cdot 10^{-5}$

$К_{ТЕР4} = 4,99 / 403406,99 = 1,24 \cdot 10^{-5}$

Як бачимо, найбільш ефективним є перший варіант реалізації програми з коефіцієнтом техніко-економічного рівня $К_{ТЕР1} = 2,38 \cdot 10^{-5}$.

4.5 Висновки

Після виконання функціонально-вартісного аналізу програмного комплексу, що розроблюється, можна зробити висновок, що з альтернатив, що залишились після першого відбору двох варіантів виконання програмного комплексу оптимальним є перший варіант реалізації програмного продукту. У нього виявився найкращий показник техніко-економічного рівня якості:

$К_{ТЕР1} = 5,88 / 246479,21 = 2,38 \cdot 10^{-5}$.

До першого варіанту реалізації відносятся такі задачі:

Мова програмування Java;

Мова запитів My SQL;

Реалізація замовлення вручну;

Інтерфейс користувача розроблений за допомогою бібліотеки Swing;

Як бачимо, даний варіант реалізації дозволяє швидко та просто реалізувати необхідні функції, а також задовольняє потреби у функціональності.

Список літератури

1. Калентьева, Ю. Н. Проблема реализации и внедрения MRP-систем на современном промышленном предприятии. URL: https://moluch.ru/archive/151/42834/ (дата звернення 20.03.20)
2. Система MRP. URL: http://www.grandars.ru/college/ekonomika-firmy/sistema-mrp.html (дата звернення 19.03.20)
3. 2007 Гаджанский А.М. Логистика. 15-е видання. Москва 2007. 467 с. (дата звернення 13.03.20)
4. Структура MRP-системы. URL: http://www.kgau.ru/istiki/isu/ch05s02.html (дата звернення 30.03.20)
5. Механизм планирования потребности в компонентах изделий при зависимом спросе. URL: http://infomanagement.ru/lekciya/Mekhanizm_planirovaniya_potrebnosti_v_komponentakh_izdelii_pri_zavisimogo_sprose (дата звернення 02.04.20)
6. Михаил Григорьев, Сергей Уваров. Логистика. 4-е изд. Москва 2014. 830 с. (дата звернення 25.03.20)
7. Эфект хлыста. URL: http://ru.scm.gsom.spbu.ru/cite_note-14 (дата звернення 28.04.20)
8. Олийнык С.В. «Операционное совершенство». ABM Cloud 2019. (дата звернення 20.04.20)
9. Щепаник Е.С.«Управление запасами по ТОС». ABM Cloud 2017. (дата звернення 15.03.20)
10. Exploration of the bull whip effect based on the evolutionary least-mean-square algorithm. URL: https://pdfs.semanticscholar.org/27dd/7fbb0ff7ebd92d32177ed4d577d8369135dc.pdf
11. Элияху Голдратт. «Критическая цепь». Альпина Паблишер. Москва 2014. 288 с. (дата звернення 26.03.20)
12. What is lean? URL: https://www.lean.org/whatslean/ (дата звернення 15.04.20)
13. Чистов Д., Амириди Ю., Кочанова У. Информационные системы в

экономике. Управление эффективностью банковского бизнеса. Москва. КноРус,2011. 174 с. (дата звернення 03.05.20)

14. Система MRP. URL: http://www.grandars.ru/college/ekonomika-firmy/sistema-mrp.html (дата звернення 15.05.20).
15. Баэрсокс Дональд Дж., Клосс Дейвид Дж. Логистика: интегрированная цепь поставок. Москва 2001. 378 с. (дата звернення 17.04.20)
16. Банько В.Г. Логістика: навчальний посібник. Київ 2013. 345 с. (дата звернення 4.05.20)
17. Гудзь П.В. Аналіз зарубіжного досвіду застосування системного управління логістичною діяльністю. Бізнесінформ 2015. 142 с. (дата звернення 8.05.20)
18. Кальченко А.Г. Логістика: підручник. Київ 2012. КНЕУ. 284 с. (дата звернення 13.04.20)
19. Мельник Н., Михайлишин Н. ТОС – теорія, що ламає стереотипи. Сталий розвиток економіки 2011. 81-83 с. (дата звернення 18.05.20)
20. Деревинский Д. Методика повышения эффективности управления бизнес-процессами на основе теории ограничений. Творчество молодых ученых 2012. 82-84 с. (дата звернення 18.05.20)
21. Чуваев А, Лямзин О. Совершенствование модели EOQ и расширение ее возможностей для управления материальными запасами предприятий в различных условиях. Теоретические поиски и предложения 2014. 299-305 с. (дата звернення 18.05.20)
22. Шевців Л.Ю. Логістичні витрати підприємства: [монографія]. Львів : Львівська політехніка 2013. 244 с. (дата звернення 06.05.20)
23. Пономарьова Ю. В., Логістика: навчальний посібник. 2-ге вид. Киъв 2013. 328 с. (дата звернення 08.05.20)

Додаток А

Склад

```
package MainFiles;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;


public class Inventory extends javax.swing.JFrame {


    public Inventory() {
        initComponents();
    }


    @SuppressWarnings("unchecked")

    private void initComponents() {

        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jScrollPane1 = new javax.swing.JScrollPane();
        table = new javax.swing.JTable();
        jButton1 = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jLabel2.setFont(new java.awt.Font("Times New Roman", 1, 48));
        jLabel2.setText("Inventory");

        jLabel3.setFont(new java.awt.Font("Ubuntu", 1, 15));

        table.setBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(0, 0, 0)));
        table.setFont(new java.awt.Font("Times New Roman", 1, 24));
        table.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
                "Item_Id", "item_name", "quantity","unit", "price"
            }
        ) {
            boolean[] canEdit = new boolean [] {
                false, false, false, false, false
            };

            public boolean isCellEditable(int rowIndex, int columnIndex) {
                return canEdit [columnIndex];
            }
        });
        table.setRowHeight(30);
        table.setRowMargin(2);
```

```
        jScrollPane1.setViewportView(table);
        if (table.getColumnModel().getColumnCount() > 0) {
            table.getColumnModel().getColumn(0).setMaxWidth(300);
            table.getColumnModel().getColumn(1).setMaxWidth(300);
            table.getColumnModel().getColumn(2).setMaxWidth(300);
            table.getColumnModel().getColumn(3).setMaxWidth(300);
            table.getColumnModel().getColumn(4).setMaxWidth(300);

        }

        jButton1.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jButton1.setText("Display Inventory");
        jButton1.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jButton1MouseClicked(evt);
            }
        });
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });

        jButton2.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jButton2.setText("Back");
        jButton2.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jButton2MouseClicked(evt);
            }
        });
        jButton2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton2ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(521, 521, 521)
                        .addComponent(jLabel3))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(379, 379, 379)
                        .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE,
295, javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(66, 66, 66)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
                            .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 1016, javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addGroup(layout.createSequentialGroup()
                                .addComponent(jButton1,
javax.swing.GroupLayout.PREFERRED_SIZE, 225, javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
```

```
                                .addComponent(jButton2,
javax.swing.GroupLayout.PREFERRED_SIZE, 95, javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addGap(13, 13, 13)))))
                .addContainerGap(31, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(25, 25, 25)
                .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 46,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(33, 33, 33)
                        .addComponent(jButton1)
                        .addGap(32, 32, 32)
                        .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 355, javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE,
30, javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(19, 19, 19)
                        .addComponent(jButton2)))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        pack();
    }

    private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {
       this.setVisible(false);
        mainFrame mm=new mainFrame();
        mm.setVisible(true);
    }

    private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {

        DefaultTableModel model = (DefaultTableModel) table.getModel();
        try {
            Class.forName("java.sql.DriverManager");
            Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/diplomaDB", "root", "");

            Statement stmt = con.createStatement();
            String query = "select * from inventory ";
            ResultSet rs=stmt.executeQuery(query);

            while(rs.next()) {
                String id = rs.getString("item_id");
                String name = rs.getString("item_name");
                String qun = rs.getString("quantity");
                String unit = rs.getString("unit");

                String price = rs.getString("price per unit");



              model.addRow(new Object[] {id,name,qun,unit,price});
```

```
            }
            rs.close();

        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, e.getMessage());

        }
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

    }


    public static void main(String args[]) {

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Inventory.class.getName()).log(java.util.logging.Lev
el.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Inventory.class.getName()).log(java.util.logging.Lev
el.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Inventory.class.getName()).log(java.util.logging.Lev
el.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Inventory.class.getName()).log(java.util.logging.Lev
el.SEVERE, null, ex);
        }



        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Inventory().setVisible(true);

            }
        });
    }


    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTable table;
```

```
    }
```

Додаток Б
Сторінка поставок

```
package MainFiles;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;



public class supplierapprove extends javax.swing.JFrame {
   String roll;
   public supplierapprove() {
      initComponents();
      roll="";
   }
   @SuppressWarnings("unchecked")

   private void initComponents() {

      jScrollPane1 = new javax.swing.JScrollPane();
      table2 = new javax.swing.JTable();
      jButton1 = new javax.swing.JButton();
      jButton2 = new javax.swing.JButton();
      jLabel1 = new javax.swing.JLabel();
      jLabel2 = new javax.swing.JLabel();
      jButton3 = new javax.swing.JButton();

      setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

      table2.setFont(new java.awt.Font("Times New Roman", 1, 24));
      table2.setModel(new javax.swing.table.DefaultTableModel(
```

```
    new Object [][] {},
    new String [] {
      "creator", "item_id", "item_name", "quantity", "unit price", "total price"
    }) {
    boolean[] canEdit = new boolean [] {
      false, false, false, false, false, false
    };

    public boolean isCellEditable(int rowIndex, int columnIndex) {
      return canEdit [columnIndex];
    }
});
table2.setRowHeight(25);
table2.setRowMargin(3);
jScrollPane1.setViewportView(table2);

jButton1.setFont(new java.awt.Font("Times New Roman", 1, 24));
jButton1.setText("view orders");
jButton1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
      jButton1ActionPerformed(evt);
    }
});

jButton2.setFont(new java.awt.Font("Times New Roman", 1, 24));
jButton2.setText("Logout");
jButton2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
      jButton2ActionPerformed(evt);
    }
});

jLabel1.setFont(new java.awt.Font("Times New Roman", 1, 24));
jLabel1.setText("Total Amount");

jLabel2.setFont(new java.awt.Font("Times New Roman", 1, 24));
```

```
        jButton3.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jButton3.setText("Approve order");
        jButton3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton3ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                .addGap(39, 39, 39)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jLabel1)
                        .addGap(18, 18, 18)
                        .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 103,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 298,
Short.MAX_VALUE)
                        .addComponent(jButton3))
                    .addComponent(jScrollPane1, javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()
                        .addComponent(jButton2)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jButton1)))
                .addGap(66, 66, 66))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(11, 11, 11)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
```

```
                .addComponent(jButton1)
                .addComponent(jButton2))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(18, 18, 18)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                .addComponent(jButton3)
                .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                    .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 27,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel1)))
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

        pack();
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {



        DefaultTableModel model = (DefaultTableModel) table2.getModel();
        model.setRowCount(0);
        try {
            Class.forName("java.sql.DriverManager");
            Connection con = (Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/diplomaDB",
"root", "");

            Statement stmt = con.createStatement();
            String query = "select * from purchase ";
            ResultSet rs=stmt.executeQuery(query);

            while(rs.next()) {
              String creator = rs.getString("creator");
                String id = rs.getString("item_id");
                String name = rs.getString("item_name");
```

```
            String qun = rs.getString("quantity");
             String price1 = rs.getString("price_per_unit");

            String totprice = rs.getString("total_price");



          model.addRow(new Object[] {creator,id,name,qun,price1,totprice});
        }

        double sum=0;
        for(int i=0;i<table2.getRowCount();i++)
        {
       sum=sum+Double.parseDouble(table2.getValueAt(i,5).toString());
        }
    jLabel2.setText(Double.toString(sum));

    rs.close();
     }catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());

     }

  }

  private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
this.setVisible(false);
    Slogin mm=new Slogin();
     mm.setVisible(true);
  }

  private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
     workWithDatabase();
    jButton1ActionPerformed(evt);

  }
   public void workWithDatabase()
```

```
  {
      Connection c=null;
       Statement  s=null;
       ResultSet rs1=null;
       try{

      Class.forName("com.mysql.jdbc.Driver");
     c=DriverManager.getConnection("jdbc:mysql://localhost/diplomaDB","root","");
     s=c.createStatement();

 DefaultTableModel model = (DefaultTableModel) table2.getModel();
         int selectedRowIndex =table2.getSelectedRow();
        String a=model.getValueAt(selectedRowIndex,1).toString();

          String c1=model.getValueAt(selectedRowIndex,3).toString();
           String s1=a;
           double id;

      if (c1!="") {
        id=Double.parseDouble(c1);}
      else {
            id = 0;
  }
   rs1 = s.executeQuery ("select quantity from  inventory where item_id="+"'"+s1+"'");
          rs1.next();
           Double id2 = rs1.getDouble("quantity");
            double id3=id2+id;
           s.executeUpdate("Update inventory set quantity="+id3+" where item_id="+"'"+s1+"'");
           s.executeUpdate("Delete from purchase where item_id ="+"'"+s1+"'");
          rs1.close();
}
 catch(SQLException | ClassNotFoundException e1)
       {
          System.out.println(e1);
       }
```

```
    }
    public static void main(String args[]) {

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger(supplierapprove.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger(supplierapprove.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(supplierapprove.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(supplierapprove.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
        }
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new supplierapprove().setVisible(true);
            }
        });
    }
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTable table2;

}
```

Додаток В
Пошук та замовлення ресурсу

```
package MainFiles;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import javax.swing.JOptionPane;

import javax.swing.table.DefaultTableModel;


public class Purchase extends javax.swing.JFrame {

    public Purchase() {

        initComponents();

    }

    @SuppressWarnings("unchecked")


    private void initComponents() {


        jLabel2 = new javax.swing.JLabel();

        bid = new javax.swing.JTextField();

        jScrollPane1 = new javax.swing.JScrollPane();

        table = new javax.swing.JTable();

        jButton1 = new javax.swing.JButton();

        jButton2 = new javax.swing.JButton();

        jLabel4 = new javax.swing.JLabel();

        jLabel5 = new javax.swing.JLabel();

        jLabel6 = new javax.swing.JLabel();

        jLabel8 = new javax.swing.JLabel();

        jLabel9 = new javax.swing.JLabel();

        jTextField1 = new javax.swing.JTextField();

        jTextField2 = new javax.swing.JTextField();

        jTextField3 = new javax.swing.JTextField();

        jTextField4 = new javax.swing.JTextField();

        jTextField5 = new javax.swing.JTextField();

        jButton3 = new javax.swing.JButton();
```

```
        jLabel1 = new javax.swing.JLabel();
        jButton5 = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setBackground(new java.awt.Color(0, 255, 0));

        jLabel2.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jLabel2.setText("Search Product By Id");

        bid.setFont(new java.awt.Font("Times New Roman", 1, 24));
        table.setFont(new java.awt.Font("Times New Roman", 1, 24));
        table.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
                "item_id", "item_name", "quntatity", "price"
            }
        ) {
            boolean[] canEdit = new boolean [] {
                false, false, false, false
            };

            public boolean isCellEditable(int rowIndex, int columnIndex) {
                return canEdit [columnIndex];
            }
        });
        table.setRowHeight(25);
        table.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                tableMouseClicked(evt);
            }
        });
        jScrollPane1.setViewportView(table);

        jButton1.setFont(jButton1.getFont().deriveFont(jButton1.getFont().getStyle() | java.awt.Font.BOLD,
jButton1.getFont().getSize()+11));
```

```
        jButton1.setText("Search");
        jButton1.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
        jButton1.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jButton1MouseClicked(evt);
            }
        });
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });

        jButton2.setFont(jButton2.getFont().deriveFont(jButton2.getFont().getStyle() | java.awt.Font.BOLD,
jButton2.getFont().getSize()+11));
        jButton2.setText("Back");
        jButton2.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
        jButton2.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jButton2MouseClicked(evt);
            }
        });

        jLabel4.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jLabel4.setText("Item_Id:");

        jLabel5.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jLabel5.setText("Item_Name:");

        jLabel6.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jLabel6.setText("Total Price:");

        jLabel8.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jLabel8.setText("Qunatity (in lit/kg):");

        jLabel9.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jLabel9.setText("Price Per Unit:");
```

```
        jTextField1.setFont(jTextField1.getFont().deriveFont(jTextField1.getFont().getStyle() | java.awt.Font.BOLD, jTextField1.getFont().getSize()+11));
        jTextField1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jTextField1ActionPerformed(evt);
            }
        });

        jTextField2.setFont(jTextField2.getFont().deriveFont(jTextField2.getFont().getStyle() | java.awt.Font.BOLD, jTextField2.getFont().getSize()+11));

        jTextField3.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jTextField3.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jTextField3MouseClicked(evt);
            }
            public void mouseEntered(java.awt.event.MouseEvent evt) {
                jTextField3MouseEntered(evt);
            }
            public void mouseExited(java.awt.event.MouseEvent evt) {
                jTextField3MouseExited(evt);
            }
        });
        jTextField3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jTextField3ActionPerformed(evt);
            }
        });

        jTextField4.setFont(new java.awt.Font("Times New Roman", 1, 24));

        jTextField5.setFont(new java.awt.Font("Times New Roman", 1, 24));

        jButton3.setFont(jButton3.getFont().deriveFont(jButton3.getFont().getStyle() | java.awt.Font.BOLD, jButton3.getFont().getSize()+11));
        jButton3.setText("Purchase");
```

```
        jButton3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton3ActionPerformed(evt);
            }
        });

        jLabel1.setFont(new java.awt.Font("Times New Roman", 1, 36));
        jLabel1.setText("Purchase");

        jButton5.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jButton5.setText("Purchase Return");
        jButton5.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton5ActionPerformed(evt);
            }
        });
        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(48, 48, 48)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jLabel8)
                            .addComponent(jLabel9)
                            .addComponent(jLabel6))
                        .addGap(27, 27, 27)
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
                            .addComponent(jTextField4, javax.swing.GroupLayout.DEFAULT_SIZE, 177, Short.MAX_VALUE)
                            .addComponent(jTextField5))
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 748,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
```

```
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(50, 50, 50)
                        .addComponent(jButton3)
                        .addGap(63, 63, 63)
                        .addComponent(jButton5))
                    .addGroup(layout.createSequentialGroup()
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                            .addComponent(jTextField3, javax.swing.GroupLayout.PREFERRED_SIZE, 177,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addGroup(layout.createSequentialGroup()
                                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                    .addComponent(jLabel4)
                                    .addComponent(jLabel5))
                                .addGap(103, 103, 103)
                                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                    .addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED_SIZE, 177,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED_SIZE, 177,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                        .addGap(241, 241, 241)
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 117,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addGroup(layout.createSequentialGroup()
                                .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 238,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addGap(42, 42, 42)
                                .addComponent(bid, javax.swing.GroupLayout.PREFERRED_SIZE, 169,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(417, 417, 417)
                        .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 163,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(571, 571, 571)
                        .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 95,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGap(0, 0, Short.MAX_VALUE)))
            .addContainerGap())
```

```
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addContainerGap()
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                            .addComponent(jButton2)
                            .addComponent(jLabel1))
                        .addGap(34, 34, 34)
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jLabel2)
                            .addComponent(bid, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(59, 59, 59)
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jLabel4)
                            .addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))))
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(20, 20, 20)
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                            .addComponent(jLabel5)
                            .addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(40, 40, 40)
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                            .addComponent(jLabel8)
                            .addComponent(jTextField3, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(48, 48, 48)
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                            .addComponent(jTextField4, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jLabel9))
```

```
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(60, 60, 60)
                        .addComponent(jLabel6)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 92,
Short.MAX_VALUE))
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 60,
Short.MAX_VALUE)
                        .addComponent(jTextField5, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(86, 86, 86)))
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jButton3)
                    .addComponent(jButton5, javax.swing.GroupLayout.PREFERRED_SIZE, 37,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(48, 48, 48))
            .addGroup(layout.createSequentialGroup()
                .addGap(36, 36, 36)
                .addComponent(jButton1)
                .addGap(33, 33, 33)
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 277,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(0, 0, Short.MAX_VALUE))))
        );

        pack();
    }

    private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
            String ii=bid.getText();
        DefaultTableModel model = (DefaultTableModel) table.getModel();
          try {
            Class.forName("java.sql.DriverManager");
            Connection con = (Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/diplomaDB",
"root", "");

            Statement stmt = con.createStatement();
```

```
            String query = "select * from inventory where item_id='"+ii+"' ";
            ResultSet rs=stmt.executeQuery(query);

            while(rs.next()) {
                String id = rs.getString("item_id");
                String name = rs.getString("item_name");
                String qun = rs.getString("quantity");
                 String price = rs.getString("price per unit");

               model.addRow(new Object[] {id,name,qun,price});
            }
            rs.close();

        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, e.getMessage());

        }

    }

    private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {
       this.setVisible(false);
        mainFrame mm=new mainFrame();
        mm.setVisible(true);
    }
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
jTextField1.setText("0");
jTextField2.setText("0");
jTextField3.setText("0");
jTextField4.setText("0");
jTextField5.setText("0");
    }

    private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {
    }
```

```
private void tableMouseClicked(java.awt.event.MouseEvent evt) {
       DefaultTableModel model=(DefaultTableModel)table.getModel();
       int selectedRowIndex =table.getSelectedRow();
       jTextField1.setText(model.getValueAt(selectedRowIndex,0).toString());
      jTextField2.setText(model.getValueAt(selectedRowIndex,1).toString());

       jTextField4.setText(model.getValueAt(selectedRowIndex,3).toString());

}

private void jTextField3MouseEntered(java.awt.event.MouseEvent evt) {
}

private void jTextField3MouseClicked(java.awt.event.MouseEvent evt) {
}

private void jTextField3MouseExited(java.awt.event.MouseEvent evt) {
   String qun= jTextField3.getText();
  String pri=jTextField4.getText();
 int N1;
 double N3;
  if (qun !="" && pri !="") {
           N3=Double.parseDouble(pri);
      N1=Integer.parseInt(qun);
 }else {
          N3=0;
          N1=0;
 }
  double N4 = N1*N3;
    jTextField5.setText(String.valueOf(N4));
}

private void jTextField3ActionPerformed(java.awt.event.ActionEvent evt) {

}
```

```
    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        String item_id=jTextField1.getText();
        String item_name=jTextField2.getText();
        String quantity=jTextField3.getText();
        String price=jTextField4.getText();
        String totprice = jTextField5.getText();

        try{
            Class.forName("com.mysql.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/diplomaDB","root","");
            String query = "insert into purchase(creator,item_id,item_name,quantity,price_per_unit,total_price)
values('user','"+item_id+"','"+item_name+"','"+quantity+"','"+price+"','"+totprice+"')";
            Statement smt3=con.createStatement();
                    int success=smt3.executeUpdate(query);
                        if(success==1)
                        {
                            JOptionPane.showMessageDialog(this, "Product Successfully Requested ");
                        }
                        else
                        {
                            JOptionPane.showMessageDialog(this, "Problem in Saving. Retry");
                        }
        }

    catch(Exception e)
        {
           JOptionPane.showMessageDialog(this, e.getMessage());
        }

    }
    private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
      workWithDatabase();
}
         public void workWithDatabase()
    {
        Connection c=null;
         Statement  s=null;
```

```
        ResultSet rs1=null;
        //int flag=0;
        try{
       Class.forName("com.mysql.jdbc.Driver");
      c=DriverManager.getConnection("jdbc:mysql://localhost/diplomaDB","root","");
      s=c.createStatement();
 DefaultTableModel model = (DefaultTableModel) table.getModel();
         int selectedRowIndex =table.getSelectedRow();
        String a=model.getValueAt(selectedRowIndex,0).toString();
          String c1=jTextField3.getText();
           String s1=a;
       int id=Integer.parseInt(c1);
   rs1 = s.executeQuery ("select quantity from  inventory where item_id="+"'"+s1+"'");
           while(rs1.next()) {
               String id1=rs1.getString("quantity");
               int id2=Integer.parseInt(id1);
              int id3=id2-id;
            s.executeUpdate("Update inventory set quantity="+id3+" where item_id="+"'"+s1+"'");
         }
          rs1.close();
}
 catch(SQLException | ClassNotFoundException e1)
       {
          System.out.println(e1);
       }
   }
   public static void main(String args[]) {
      try {
         for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
               javax.swing.UIManager.setLookAndFeel(info.getClassName());
               break;
            }
         }
      } catch (ClassNotFoundException ex) {
         java.util.logging.Logger.getLogger(Purchase.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
```

```
        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger(Purchase.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(Purchase.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(Purchase.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        }
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Purchase().setVisible(true);
            }
        });
    }
    private javax.swing.JTextField bid;
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    private javax.swing.JButton jButton5;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTextField jTextField1;
    private javax.swing.JTextField jTextField2;
    private javax.swing.JTextField jTextField3;
    private javax.swing.JTextField jTextField4;
    private javax.swing.JTextField jTextField5;
    private javax.swing.JTable table;

}
```

Додаток Г
Пошук продукту

```
package MainFiles;


import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
public class Searchproduct extends javax.swing.JFrame {
    public Searchproduct() {
        initComponents();
    }
    @SuppressWarnings("unchecked")
    private void initComponents() {
        jLabel2 = new javax.swing.JLabel();
        bid = new javax.swing.JTextField();
        jScrollPane1 = new javax.swing.JScrollPane();
        table = new javax.swing.JTable();
        jButton1 = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();
        jLabel4 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        jLabel6 = new javax.swing.JLabel();
        jLabel8 = new javax.swing.JLabel();
        jLabel9 = new javax.swing.JLabel();
        jTextField1 = new javax.swing.JTextField();
        jTextField2 = new javax.swing.JTextField();
        jTextField3 = new javax.swing.JTextField();
        jTextField4 = new javax.swing.JTextField();
        jTextField5 = new javax.swing.JTextField();
        jButton3 = new javax.swing.JButton();
        jButton4 = new javax.swing.JButton();
        jLabel1 = new javax.swing.JLabel();
```

```
setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
setBackground(new java.awt.Color(51, 255, 0));
setFont(new java.awt.Font("Times New Roman", 1, 36));
setForeground(new java.awt.Color(0, 255, 0));
jLabel2.setFont(new java.awt.Font("Times New Roman", 1, 24));
jLabel2.setText("Search Product");
bid.setFont(new java.awt.Font("Times New Roman", 1, 24));
table.setFont(new java.awt.Font("Times New Roman", 1, 24));
table.setModel(new javax.swing.table.DefaultTableModel(
    new Object [][] {
    },
    new String [] {
        "p_id", "p_name", "pack","price"
    }
) {
    boolean[] canEdit = new boolean [] {
        false, false, false, false
    };
    public boolean isCellEditable(int rowIndex, int columnIndex) {
        return canEdit [columnIndex];
    }
});
table.setRowHeight(25);
table.setRowMargin(2);
table.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        tableMouseClicked(evt);
    }
});
jScrollPane1.setViewportView(table);
if (table.getColumnModel().getColumnCount() > 0) {
    table.getColumnModel().getColumn(0).setMinWidth(120);
    table.getColumnModel().getColumn(1).setMinWidth(20);
    table.getColumnModel().getColumn(2).setMinWidth(140);
    table.getColumnModel().getColumn(3).setMinWidth(150);
```

```
        }

        jButton1.setFont(jButton1.getFont().deriveFont(jButton1.getFont().getStyle() | java.awt.Font.BOLD,
jButton1.getFont().getSize()+11));
       //  jButton1.setIcon(new javax.swing.ImageIcon("C:\\Users\\kau9f\\Documents\\Teletubbies.jpg"));
        jButton1.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
        jButton1.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jButton1MouseClicked(evt);
            }
        });
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });

        jButton2.setFont(jButton2.getFont().deriveFont(jButton2.getFont().getStyle() | java.awt.Font.BOLD,
jButton2.getFont().getSize()+11));
        jButton2.setText("Back");
        jButton2.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
        jButton2.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jButton2MouseClicked(evt);
            }
        });
        jButton2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton2ActionPerformed(evt);
            }
        });

        jLabel4.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jLabel4.setText("Product Id:");

        jLabel5.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jLabel5.setText("Product Name:");
```

```
        jLabel6.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jLabel6.setText("Total Price:");

        jLabel8.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jLabel8.setText("Qunatity of pack:");

        jLabel9.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jLabel9.setText("Price Per Pack:");

        jTextField1.setFont(jTextField1.getFont().deriveFont(jTextField1.getFont().getStyle() | java.awt.Font.BOLD,
jTextField1.getFont().getSize()+11));
        jTextField1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jTextField1ActionPerformed(evt);
            }
        });

        jTextField2.setFont(jTextField2.getFont().deriveFont(jTextField2.getFont().getStyle() | java.awt.Font.BOLD,
jTextField2.getFont().getSize()+11));

        jTextField3.setFont(new java.awt.Font("Times New Roman", 1, 24));
        jTextField3.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jTextField3MouseClicked(evt);
            }
            public void mouseEntered(java.awt.event.MouseEvent evt) {
                jTextField3MouseEntered(evt);
            }
            public void mouseExited(java.awt.event.MouseEvent evt) {
                jTextField3MouseExited(evt);
            }
        });
        jTextField3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jTextField3ActionPerformed(evt);
            }
```

```
        });

        jTextField4.setFont(new java.awt.Font("Times New Roman", 1, 24));

        jTextField5.setFont(new java.awt.Font("Times New Roman", 1, 24));

        jButton3.setFont(jButton3.getFont().deriveFont(jButton3.getFont().getStyle() | java.awt.Font.BOLD,
jButton3.getFont().getSize()+11));
        jButton3.setText("Add to cart");
        jButton3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton3ActionPerformed(evt);
            }
        });

        jButton4.setFont(new java.awt.Font("Times New Roman", 1, 36));
        jButton4.setText("View Cart");
        jButton4.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton4ActionPerformed(evt);
            }
        });

        jLabel1.setFont(new java.awt.Font("Times New Roman", 1, 36));
        jLabel1.setText("Search Product");
        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 95,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
```

```
                .addComponent(jLabel1)
                .addGap(331, 331, 331)
                .addComponent(jButton4, javax.swing.GroupLayout.PREFERRED_SIZE, 236,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(31, 31, 31))
            .addGroup(layout.createSequentialGroup()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
                            .addGroup(layout.createSequentialGroup()
                                .addComponent(jLabel6)
                                .addGap(155, 155, 155)
                                .addComponent(jTextField5))
                            .addGroup(layout.createSequentialGroup()
                                .addComponent(jLabel5)
                                .addGap(152, 152, 152)
                                .addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED_SIZE, 158,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGroup(layout.createSequentialGroup()
                                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                    .addComponent(jLabel9)
                                    .addComponent(jLabel8))
                                .addGap(76, 76, 76)
                                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
                                    .addComponent(jTextField3)
                                    .addComponent(jTextField4, javax.swing.GroupLayout.PREFERRED_SIZE, 158,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                                .addGap(10, 10, 10)
                                .addComponent(jLabel4)
                                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED_SIZE, 158,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addGroup(layout.createSequentialGroup()
                                .addGap(18, 18, 18)
                                .addComponent(jScrollPane1))
```

```
                        .addGroup(layout.createSequentialGroup()
                            .addGap(162, 162, 162)
                            .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 195,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addGap(35, 35, 35)
                            .addComponent(bid, javax.swing.GroupLayout.PREFERRED_SIZE, 149,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                            .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 29,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addGap(0, 195, Short.MAX_VALUE))))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(136, 136, 136)
                        .addComponent(jButton3)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 900,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addContainerGap()))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(52, 52, 52)
                        .addComponent(jLabel1))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(27, 27, 27)
                        .addComponent(jButton2))
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(jButton4, javax.swing.GroupLayout.PREFERRED_SIZE, 66,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGap(99, 99, 99)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                            .addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                    .addComponent(jLabel4))
                .addGap(35, 35, 35))
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jButton1)
                    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                        .addComponent(jLabel2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(bid, javax.swing.GroupLayout.PREFERRED_SIZE, 37,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGap(48, 48, 48)))
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel5)
                    .addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED_SIZE, 48,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(20, 20, 20)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel8)
                    .addComponent(jTextField3, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(26, 26, 26)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel9)
                    .addComponent(jTextField4, javax.swing.GroupLayout.PREFERRED_SIZE, 43,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel6)
                    .addComponent(jTextField5, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(51, 51, 51))
            .addGroup(layout.createSequentialGroup()
                .addGap(22, 22, 22)
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 265,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
```

```
            .addComponent(jButton3)
            .addGap(207, 207, 207)) );
        pack();
    }
    private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
          String ii=bid.getText();
       DefaultTableModel model = (DefaultTableModel) table.getModel();
         try {
          Class.forName("java.sql.DriverManager");
          Connection con = (Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/diplomaDB",
"root", "");
          Statement stmt = con.createStatement();
           String query = "select * from products where p_id='"+ii+"' ";
          ResultSet rs=stmt.executeQuery(query);
          while(rs.next()) {
              String id = rs.getString("p_id");
              String name = rs.getString("p_name");
              String pack = rs.getString("pack");
               String price = rs.getString("price");
             model.addRow(new Object[] {id,name,pack,price});
          }
          rs.close();
        } catch (Exception e) {
          JOptionPane.showMessageDialog(this, e.getMessage());
        }
    }
    private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {
      this.setVisible(false);
       mainFrame mm=new mainFrame();
       mm.setVisible(true);
    }
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
jTextField1.setText("0");
jTextField2.setText("0");
jTextField3.setText("0");
jTextField4.setText("0");
jTextField5.setText("0");
```

```
}

private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {
}
private void tableMouseClicked(java.awt.event.MouseEvent evt) {
      DefaultTableModel model=(DefaultTableModel)table.getModel();
      int selectedRowIndex =table.getSelectedRow();
      jTextField1.setText(model.getValueAt(selectedRowIndex,0).toString());
     jTextField2.setText(model.getValueAt(selectedRowIndex,1).toString());
      jTextField4.setText(model.getValueAt(selectedRowIndex,3).toString());
}
private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
this.setVisible(false);
     addtocart bc = new addtocart();
     bc.setVisible(true);
}
private void jTextField3MouseEntered(java.awt.event.MouseEvent evt) {
}
private void jTextField3MouseClicked(java.awt.event.MouseEvent evt) {
}
private void jTextField3MouseExited(java.awt.event.MouseEvent evt) {
   String qun= jTextField3.getText();
  String pri=jTextField4.getText();
  double N3=Double.parseDouble(pri);
  int N1=Integer.parseInt(qun);
   // int N2=N1;
    double N4=N1*N3;
   jTextField5.setText(String.valueOf(N4));
}
private void jTextField3ActionPerformed(java.awt.event.ActionEvent evt) {
}
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    String p_id=jTextField1.getText();
    String p_name=jTextField2.getText();
    String quantity=jTextField3.getText();
```

```
        //String price=jTextField4.getText();
        String totprice = jTextField5.getText();
        try{
            Class.forName("com.mysql.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/diplomaDB","root","");
            String query = "insert into cart(item_id,item_name,quantity,total_price)
values('"+p_id+"','"+p_name+"','"+quantity+"','"+totprice+"')";
            Statement smt3=con.createStatement();
                  int success=smt3.executeUpdate(query);
                     if(success==1)
                     {
                        JOptionPane.showMessageDialog(this, "Product Successfully Added to cart ");
                     }
                     else
                     {
                        JOptionPane.showMessageDialog(this, "Problem in Saving. Retry");
                     }
        }
    catch(Exception e)
        {
           JOptionPane.showMessageDialog(this, e.getMessage());
        }
    }
    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    }
    public static void main(String args[]) {
        try {
           for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
              if ("Nimbus".equals(info.getName())) {
                 javax.swing.UIManager.setLookAndFeel(info.getClassName());
                 break;
              }
           }
        } catch (ClassNotFoundException ex) {
           java.util.logging.Logger.getLogger(Searchproduct.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
        } catch (InstantiationException ex) {
```

```
            java.util.logging.Logger.getLogger(Searchproduct.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(Searchproduct.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(Searchproduct.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
        }
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Searchproduct().setVisible(true);
            }
        });
    }
    private javax.swing.JTextField bid;
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    private javax.swing.JButton jButton4;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTextField jTextField1;
    private javax.swing.JTextField jTextField2;
    private javax.swing.JTextField jTextField3;
    private javax.swing.JTextField jTextField4;
    private javax.swing.JTextField jTextField5;
    private javax.swing.JTable table;

}
```

Додаток Ґ

Картка замовлення

```
package MainFiles;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.Date;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

public class addtocart extends javax.swing.JFrame {
   String roll;
   public addtocart() {
      initComponents();
      roll="";
   }
   @SuppressWarnings("unchecked")

   private void initComponents() {
      jScrollPane1 = new javax.swing.JScrollPane();
      table2 = new javax.swing.JTable();
      jButton1 = new javax.swing.JButton();
      jButton2 = new javax.swing.JButton();
      jLabel1 = new javax.swing.JLabel();
      jLabel2 = new javax.swing.JLabel();
      jButton3 = new javax.swing.JButton();

      setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

      table2.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N
      table2.setModel(new javax.swing.table.DefaultTableModel(
         new Object [][] {

         },
         new String [] {
            "item_id", "item_name", "quantity", "total price"
         }
      ) {
         boolean[] canEdit = new boolean [] {
             false, false, false, false
```

```
    };

    public boolean isCellEditable(int rowIndex, int columnIndex) {
        return canEdit [columnIndex];
    }
});
table2.setRowHeight(25);
table2.setRowMargin(3);
jScrollPane1.setViewportView(table2);
if (table2.getColumnModel().getColumnCount() > 0) {
    table2.getColumnModel().getColumn(1).setMinWidth(140);
}

jButton1.setFont(new java.awt.Font("Times New Roman", 1, 36)); // NOI18N
jButton1.setText("View Cart");
jButton1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton1ActionPerformed(evt);
    }
});

jButton2.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N
jButton2.setText("Back");
jButton2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton2ActionPerformed(evt);
    }
});

jLabel1.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N
jLabel1.setText("Total Amount");

jLabel2.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N

jButton3.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N
jButton3.setText("Proceed to bill");
jButton3.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton3ActionPerformed(evt);
    }
});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
```

```
 getContentPane().setLayout(layout);
 layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
       .addGap(0, 46, Short.MAX_VALUE)
       .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)
          .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 1003,
javax.swing.GroupLayout.PREFERRED_SIZE)
          .addGroup(layout.createSequentialGroup()
             .addComponent(jButton2)
             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
             .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 227,
javax.swing.GroupLayout.PREFERRED_SIZE)))
       .addGap(34, 34, 34))
    .addGroup(layout.createSequentialGroup()
       .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
          .addGroup(layout.createSequentialGroup()
             .addGap(245, 245, 245)
             .addComponent(jLabel1)
             .addGap(42, 42, 42)
             .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 103,
javax.swing.GroupLayout.PREFERRED_SIZE))
          .addGroup(layout.createSequentialGroup()
             .addGap(394, 394, 394)
             .addComponent(jButton3)))
       .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
 );
 layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
       .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
          .addGroup(layout.createSequentialGroup()
             .addGap(5, 5, 5)
             .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 69,
javax.swing.GroupLayout.PREFERRED_SIZE))
          .addGroup(layout.createSequentialGroup()
             .addGap(24, 24, 24)
             .addComponent(jButton2)))
       .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
       .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
       .addGap(19, 19, 19)
```

```
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(jLabel1)
                .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 27,
    javax.swing.GroupLayout.PREFERRED_SIZE))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 43, Short.MAX_VALUE)
            .addComponent(jButton3)
            .addGap(28, 28, 28))
    );
    pack();
}

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    DefaultTableModel model = (DefaultTableModel) table2.getModel();
    model.setRowCount(0);
   // int  total = 0;
    //String tot = "";


    try {
        Class.forName("java.sql.DriverManager");
        Connection con = (Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/diplomaDB",
    "root", "");

        Statement stmt = con.createStatement();
        String query = "select * from cart ";
        ResultSet rs=stmt.executeQuery(query);

        while(rs.next()) {
            String id = rs.getString("item_id");
            String name = rs.getString("item_name");
            String qun = rs.getString("quantity");


            String price = rs.getString("total_price");


          model.addRow(new Object[] {id,name,qun,price});
        }

        double sum=0;
        for(int i=0;i<table2.getRowCount();i++)
        {
       sum=sum+Double.parseDouble(table2.getValueAt(i,3).toString());
```

```
            }
        jLabel2.setText(Double.toString(sum));

        rs.close();


         }catch (Exception e) {
            JOptionPane.showMessageDialog(this, e.getMessage());

        }
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
this.setVisible(false);
        Searchproduct mm=new Searchproduct();
        mm.setVisible(true);
    }

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

try{DefaultTableModel model=(DefaultTableModel)table2.getModel();
            int selectedRowIndex =table2.getSelectedRow();
           //String a=model.getValueAt(selectedRowIndex,0).toString();
            String b=model.getValueAt(selectedRowIndex,0).toString();
             String c=model.getValueAt(selectedRowIndex,1).toString();
              String d=model.getValueAt(selectedRowIndex,2).toString();
              String e=model.getValueAt(selectedRowIndex,3).toString();
              //Date k = null;

    Class.forName("com.mysql.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/diplomaDB","root","");
            String query = "insert into bill(p_id,p_name,quantity,total_price) values('"+b+"','"+c+"','"+d+"','"+e+"')";
            Statement smt3=con.createStatement();
                    int success=smt3.executeUpdate(query);
                       if(success==1)
                        {
                          JOptionPane.showMessageDialog(this, "Product Successfully Added to bill ");
                        }
                        else
                        {
                           JOptionPane.showMessageDialog(this, "Problem in Saving. Retry");
                        }
```

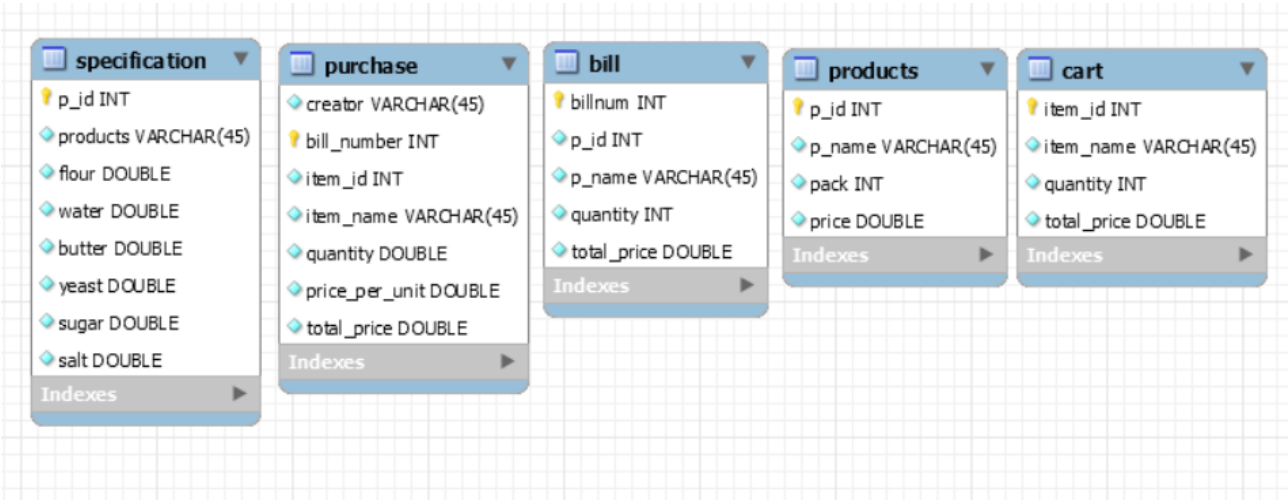
```
}
catch(Exception e)
       {
          JOptionPane.showMessageDialog(null, e);
       }
workWithDatabase();
   }
   public double[] Specification(int p_id) {
      Connection conn = null;
      Statement s1 = null;
      ResultSet rs1 = null;
      double[] sp = new double[6];
      try{

          Class.forName("com.mysql.jdbc.Driver");
         conn=DriverManager.getConnection("jdbc:mysql://localhost/diplomaDB","root","");
      s1 = conn.createStatement();
      rs1 = s1.executeQuery ("select * from  Specification where p_id="+"'"+p_id+"'");
       rs1.next();
        sp[0] = rs1.getDouble("flour");
          sp[1] = rs1.getDouble("water");
          sp[2] = rs1.getDouble("butter");
          sp[3] = rs1.getDouble("yeast");
          sp[4] = rs1.getDouble("sugar");
          sp[5] = rs1.getDouble("salt");

          rs1.close();
      }catch(SQLException | ClassNotFoundException e1)
       {
          System.out.println(e1);
       }
   return sp;
   }

   public void DBUpdating(double flour_s, double water_s, double butter_s, double yeast_s, double sugar_s, double
      salt_s, int bill_quantity) {
      Connection connection = null;
      Statement s2 = null;
      ResultSet rs2 = null;

      try{
```

```
        Class.forName("com.mysql.jdbc.Driver");
       connection=DriverManager.getConnection("jdbc:mysql://localhost/diplomaDB","root","");
    s2 = connection.createStatement(ResultSet.TYPE_FORWARD_ONLY, ResultSet.CONCUR_UPDATABLE);
    rs2 = s2.executeQuery ("select * from  inventory");
    rs2.next();
      double newFlour = rs2.getDouble("quantity") - flour_s*bill_quantity;
      rs2.updateDouble("quantity", newFlour);
      rs2.updateRow();
      rs2.next();

      double newWater = rs2.getDouble("quantity") - water_s*bill_quantity;
      rs2.updateDouble("quantity", newWater);
      rs2.updateRow();
      rs2.next();

      double newButter = rs2.getDouble("quantity") - butter_s*bill_quantity;
      rs2.updateDouble("quantity", newButter);
      rs2.updateRow();
      rs2.next();

      double newYeast = rs2.getDouble("quantity") - yeast_s*bill_quantity;
      rs2.updateDouble("quantity", newYeast);
      rs2.updateRow();
      rs2.next();

      double newSugar = rs2.getDouble("quantity") - sugar_s*bill_quantity;
      rs2.updateDouble("quantity", newSugar);
      rs2.updateRow();
      rs2.next();

      double newSalt = rs2.getDouble("quantity") - salt_s*bill_quantity;
      rs2.updateDouble("quantity", newSalt);
      rs2.updateRow();

      rs2.close();
    }catch(SQLException | ClassNotFoundException e1)
     {
        System.out.println(e1);
     }
}
```

```
    public void workWithDatabase()
   {
      Connection c=null;
       Statement s3 = null;
       ResultSet rs3 = null;
      double[] sp = new double[6];

      try{

      Class.forName("com.mysql.jdbc.Driver");
     c=DriverManager.getConnection("jdbc:mysql://localhost/diplomaDB","root","");

     s3 = c.createStatement();

     rs3 = s3.executeQuery("select * from bill");
     rs3.last();
        int product_id = rs3.getInt("p_id");
        int bill_quantity = rs3.getInt("quantity");

        sp = Specification(product_id);
     DBUpdating(sp[0], sp[1], sp[2], sp[3], sp[4], sp[5], bill_quantity);
     rs3.close();

      }
      catch(SQLException | ClassNotFoundException e1)
       {
           System.out.println(e1);
       }
       try {
          Class.forName("java.sql.DriverManager");
          Connection con = (Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/diplomaDB",
     "root", "");

          Statement stmt = con.createStatement();
          String query = "delete from cart ";
                  int success=stmt.executeUpdate(query);
       }catch (Exception e) {
          JOptionPane.showMessageDialog(this, e.getMessage());
       }

this.setVisible(false);
       bill mm=new bill();
```

```
    mm.setVisible(true);
}

public static void main(String args[]) {

    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(addtocart.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(addtocart.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(addtocart.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(addtocart.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    }

    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new addtocart().setVisible(true);
        }
    });
}
private javax.swing.JButton jButton1;
private javax.swing.JButton jButton2;
private javax.swing.JButton jButton3;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JTable table2;

  }
```

Додаток Д
Структура бази даних



Додаток Е
Презентація

Дипломна робота на тему: «Управління запасами на виробництві на основі удосконаленої методології MRP»

Виконала:
студент (-ка) IV курсу, групи КА-65
Фардман К.О.
Керівник:
Недашковська Н.І.

Актуальність

- Класична MRP-система є застарілою в наших реаліях, та не може справитися з динамічністю ринку.
- Майже кожне виробниче підприємство розробляє власну систему MRP з блоком управління запасами.

Об’єкт, предмет та мета дослідження

- Об’єкт дослідження – блок управління запасами виробництва в методології MRP.
- Предмет дослідження – сучасні методи управління запасами.
- Мета роботи – покращення MRP методології, шляхом додавання основних принципів методів TOC та Lean.

Постановка задачі

- Дослідити MRP методологію та блок управління запасами класичної MRP-системи.
- Провести порівняльний аналіз існуючих методів та підходів в управлінні запасами.
- Розробити програму, у якій замінити застарілі розрахунки блоку MRP на новітні методи управління запасами.
- Зробити якісну характеристику.

Проблематика предметної області

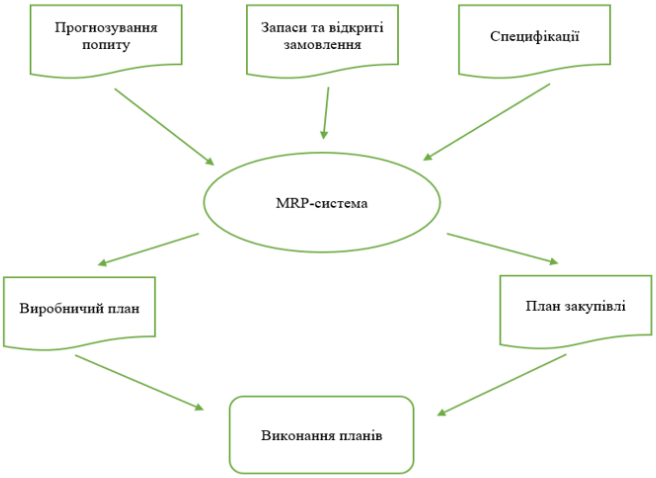
Проблеми

- Лишки
- Втрачені продажі
- Заморожені кошти

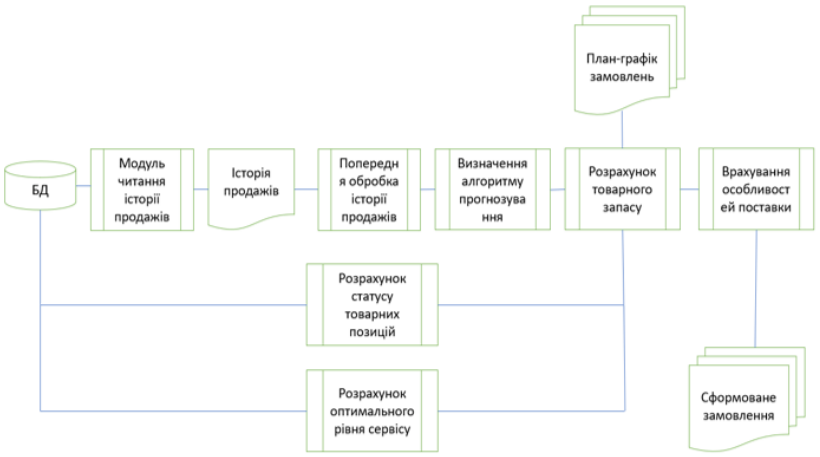
Цілі

- Безперебійна робота підприємства;
- Мінімізація запасів на складах;
- Регулювання поставок матеріалів;

Схема класичної взаємодії підприємства та MRP-системи



Структура модулю формування замовлень



Матеріальні запаси

Статуси матеріалів:

- Наявний на складі (W_i)
- Зарезервований (R_i)
- В поточних замовленнях (O_i)
- Планується замовлення (P_i)
- Страховий запас (S_i)

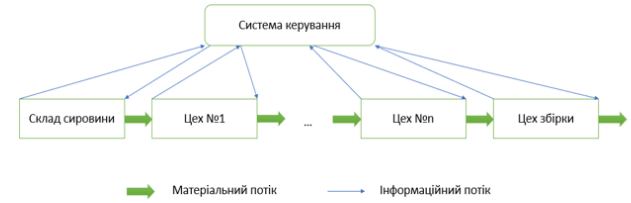
Розрахунок чистої потреби виробництва

$$N_i^p = N_i^t - W_i - S_i - R_i$$

$$N_i^p \geq 0$$

Створення нового замовлення

Керування ланцюгом поставок

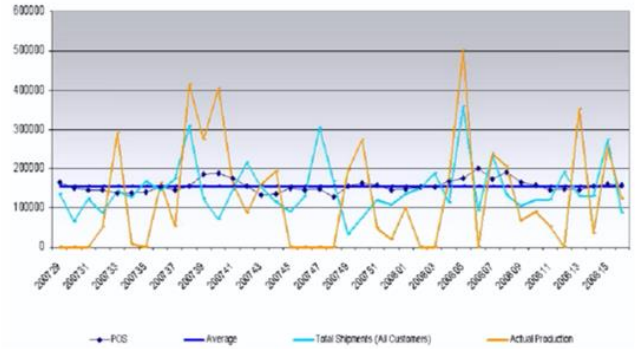


Системи виштовхування
- Засновані на складному прогнозуванні
- Класична MRP-система
- Посилює «ефект батога»

Система витягування
- Орієнтовані на збільшення гнучкості процесів виробництва
- TOC, Lean
- Руйнуються при появі дуже сильної сезонності, або введенні великої кількості нового асортименту

Ефект батога

- Ефект батога – феномен в ланцюгах поставок, який полягає в посиленні амплітуди коливання попиту (обсягу замовлень) в міру віддалення від реального джерела попиту в ланцюзі постачань.



Об'єм продажів – 25%, об'єм відвантаження – 105%, об'єм виробництва – 150%

Опис методології Теорії обмежень систем(ТОС)

Нехай процес являє собою напрямлений граф

- $P = (\Omega, A, L, U, R)$, де:
- Ω – швидкість появи екземплярів процесів.
- $A = \{a_i | i = 1, \dots, I\}$ – набір дій, які входять у екземпляр процесу.
- $L \subseteq \left\{ (a_i, a_j) \,\middle|\, a_i \in A, a_j \in A \;\&\; i \neq j \right\}$ -набір зв'язків.
- $U = \{u_k | k = 1, \dots, K\}$ – набір учасників процесу.
- $R(\subseteq A \times U) = \{(a_i, u_k) | a_i \in A, u_k \in U\}$ – набір взаємозв'язків.

Навантаження учасника:

$$WL_k = \sum_{\{a_i | (a_i, u_k) \in R\}} \frac{\lambda_{a_i} \times p_{a_i u_k}}{\mu_{a_i u_k}} = \Omega \times \sum_{\{a_i | (a_i, u_k) \in R\}} \frac{f_{a_i} \times p_{a_i u_k}}{\mu_{a_i u_k}}, \qquad \forall u_k \in U$$

Де:

- f_{a_i} – очікуваний час виконання задачі по дії a_i.
- λ_{a_i} – швидкість появи задачі по дії a_i.
- $p_{a_i u_k}$ – ймовірність розподілення задачі по дії a_i учаснику u_k.
- $\mu_{a_i u_k}$- середній час виконання учасником u_k задачі по дії a_i

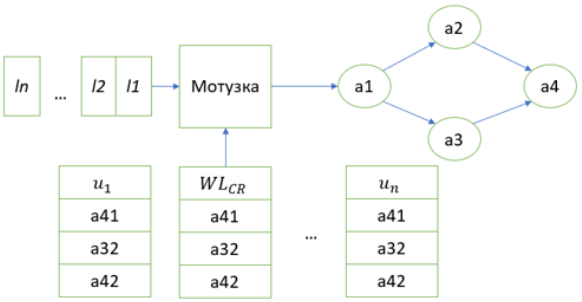
$$WL_{CR} = \max WL_k$$

Інструменти Теорії обмежень

- Барабан:

$$Drum(d) = \sum_{\{a_i | (a_i, u_k) \in R\}} \frac{1}{ET_i}$$

- Буфер
- Мотузка: $\lambda = d\left(\frac{\text{задача}}{\text{час}}\right)$



Опис методології lean – бережливого виробництва

$$C_{\text{заг}} = C_{\text{зам}} + C_{\text{зб}} \rightarrow min$$

$$C_{\text{зам}} = K \cdot \frac{N^t}{N^p}$$

$$C_{\text{зб}} = C_{\text{збо}} \cdot \frac{N^p}{2}$$

$$C_{\text{заг}} = C_{\text{зам}} + C_{\text{зб}} + C_{\text{зап}} \rightarrow min$$

За формулою Уилсона: $Q_o = \sqrt{\frac{2 \cdot N^t \cdot K}{C_{\text{збо}}}}$

$$C_{\text{заг}} = K(Q) \cdot \frac{N^t}{N^p} + C_{\text{збо}}(Q) \cdot \frac{N^p}{2} + C_{\text{запо}}(Q) \cdot N^t \rightarrow min$$

- $C_{\text{заг}}$ – загальний рівень витрат
- $C_{\text{зам}}$ – сума витрат на розміщення замовлення
- $C_{\text{зб}}$ – витрати на зберігання сировини та матеріалів
- K – вартість розміщення одного замовлення

Де:
$K(Q)$ – функція залежності вартості розміщення замовлення від його об'єму.
$C_{\text{збо}}(Q)$ – функція залежності вартості зберігання одиниці запасів від їх об'єму.
$C_{\text{запо}}(Q)$ – функція залежності вартості одиниці запасів від об'єму замовлення.

Порівняльна характеристика методів

Теорія обмежень системи

- Зосереджується на покращенні обмежень, що визначають загальну продуктивність системи.
- Значно підвищує рентабельність інвестицій та успіх програм Lean&SixSigma
- Збільшує прибуток за рахунок збільшення продажів, а не за рахунок скорочення витрат, а отже створює платформу для розширення підприємства.

Ощадливе виробництво Lean

- Найрозповсюдженіший підхід у виробництві по всьому світу.
- Сфокусований на знаходженні та зменшенні усіх форм витрат
- Багатовимірний підхід: Just-in-time, 5S, Lean Engineering,…

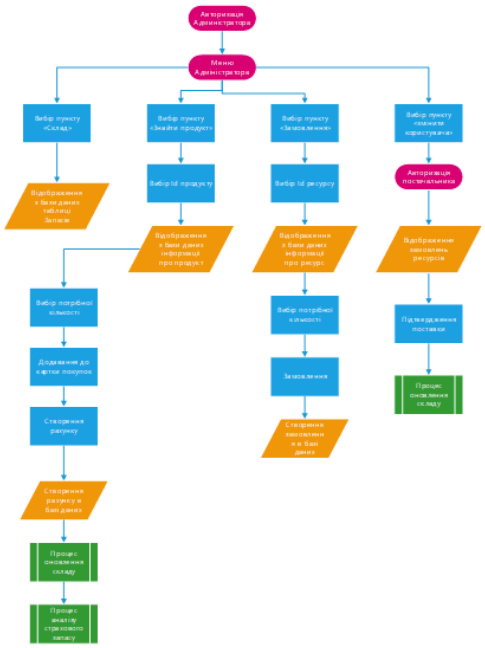
Побудова удосконаленої методології MRP

- Основною метою є поліпшення ефекту батога у класичних MRP-системах.
- Усі системи «виштовхуючого» керування ланцюгом поставок страждають від ефекту батога .
- Для подолання цього недоліку доцільно використовувати методи «витягуючих» систем.
- Для збільшення гнучкості систем пропонується використати інструмент «Буфер» з теорії обмежень систем.
- Також основною метою є мінімізація ресурсів на складі, якої можливо домогтися за допомогою принципів ощадливого виробництва.

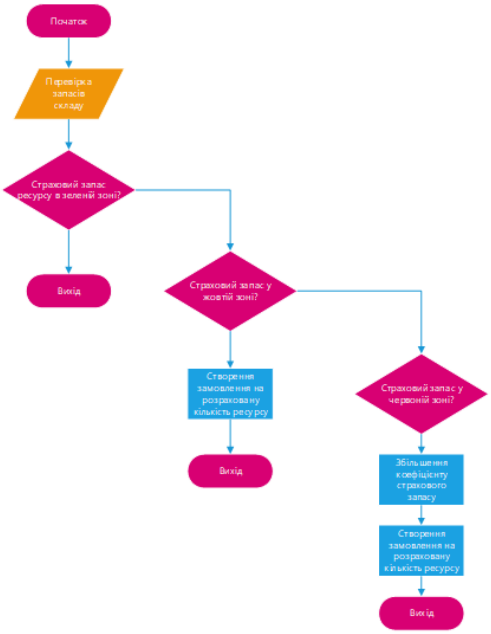
Удосконалена методологія MRP

- **Страховий запас** розраховується динамічно для окремого ресурсу підприємства, гнучко підлаштовується до середовища, сезонної активності, акцій, непередбачуваних обставин.
- **Страховий запас** ділиться на три однакові зони буферу, для автоматизованого вибору сценарію закупівлі матеріалів.

Схема удосконаленої системи MRP



Процес аналізу страхового запасу



Для кожного ресурсу розраховується страховий запас S_i :

$$S_i = \sum_{j=1}^{n} a_{ij} \cdot P_{avg} \cdot t \cdot S^p$$

Де:

P_{avg} – середній попит і-того продукту.

a_{ij} – елемент матриці специфікації і-того ресурсу j-того продукту.

t – середній час виконання замовлення.

S^p – коефіцієнт страхового запасу.

Опис розробленого програмного продукту

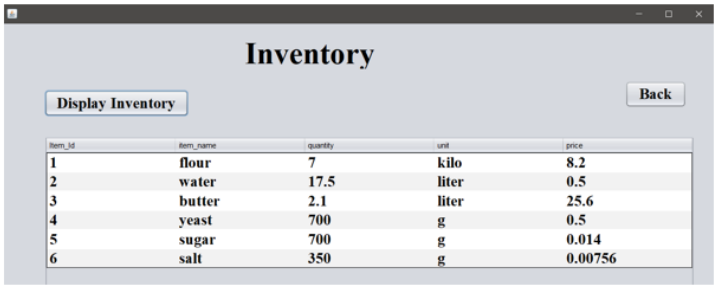
Пункти меню:

- Search Product – Знайти та замовити готовий продукт.
- Inventory – Відображення запасів матеріалів.
- Purchase – Зробити замовлення матеріалів.
- Logout – Змінити юзера (Administrator/Supplier)

У даній програмі розроблено блок управління запасами виробництва:

- Отримання замовлення на готову продукцію.
- Виділення ресурсів зі складу, згідно з специфікацією продукції.
- Автоматичне замовлення ресурсів.
- Користувацьке замовлення ресурсів.

Запаси на складі



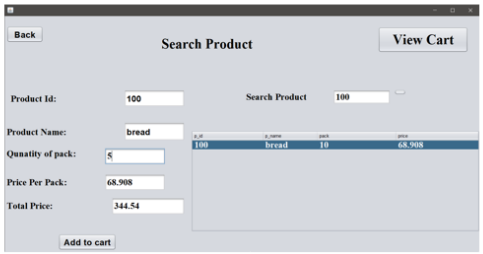
Для кожного ресурсу розраховується **страховий запас S_i** :

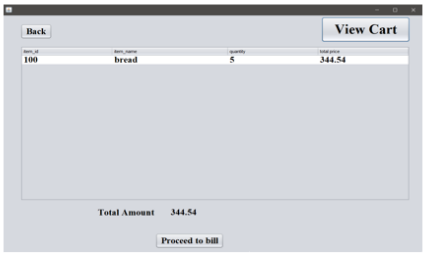
$$S_i = \sum_{j=1}^{n} a_{ij} \cdot P_{avg} \cdot t \cdot S^p$$

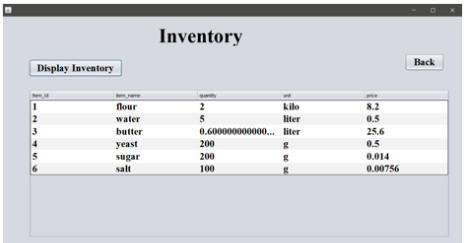
Де:
P_{avg} – середній попит і-того продукту.
a_{ij} – елемент матриці специфікації і-того ресурсу j-того продукту.
t – середній час виконання замовлення.
S^p – коефіцієнт страхового запасу.

Страховий запас ресурсу ділиться на три однакові буферні зони: зелена, жовта, червона.
Якщо кількість запасу ресурсу на складі потрапляє до жовтої зони страхового запасу, програма автоматично робить замовлення. Якщо до червоної – окрім замовлення, збільшується коефіцієнт страхового запасу, а отже і сам страховий запас.

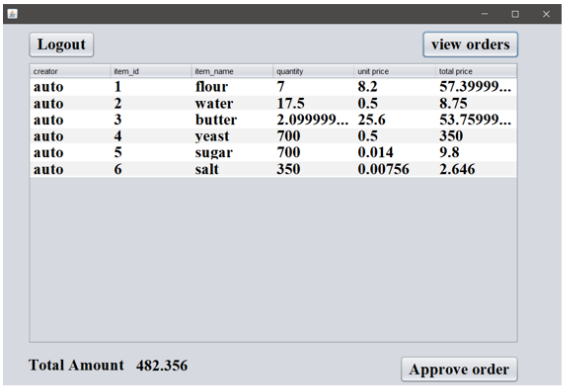
Замовлення продуктів і ресурсів

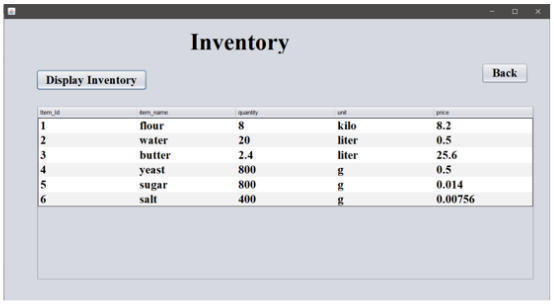




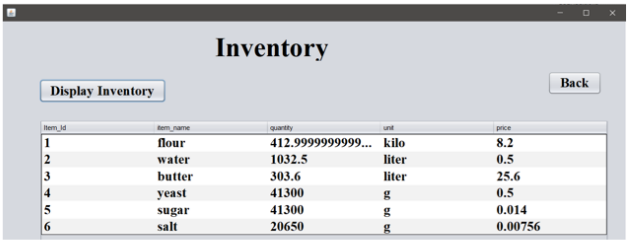


Автоматичне замовлення

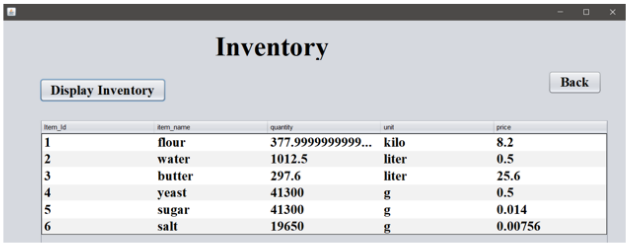




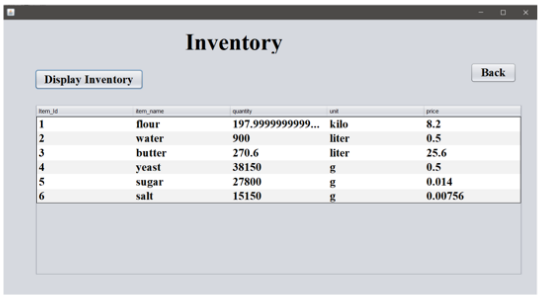
Кольорові зони буферу: зелена

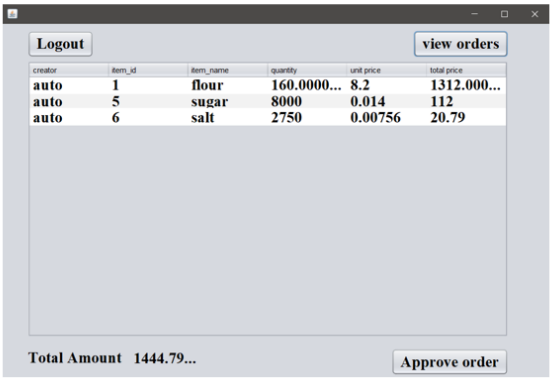


Використано ресурсів менше ніж 33% від страхового запасу. Автоматичне замовлення не сформовано.



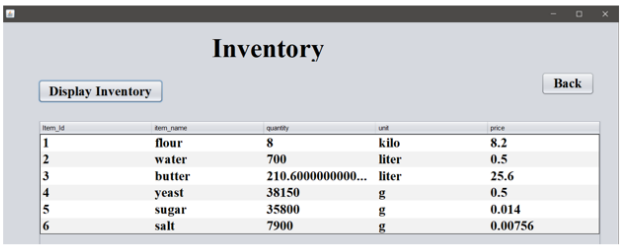
Кольорові зони буферу: жовта

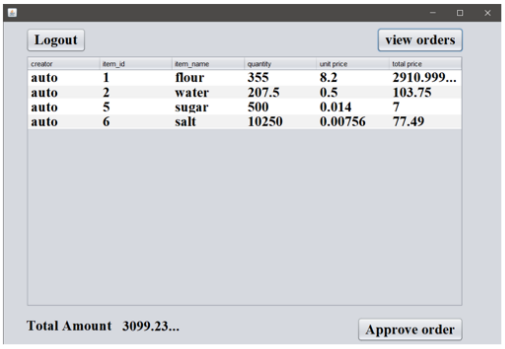




Рівень запасу 1,5,6-того ресурсів потрапив до жовтої зони. Їх витрати перевищили 33% буферу. Система сформувала автоматичне замовлення.

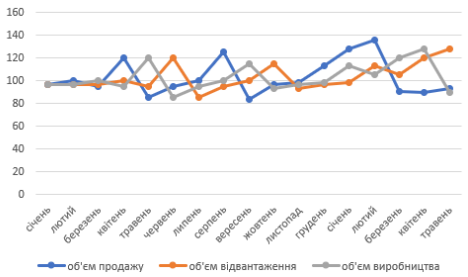
Кольорові зони буферу: червона



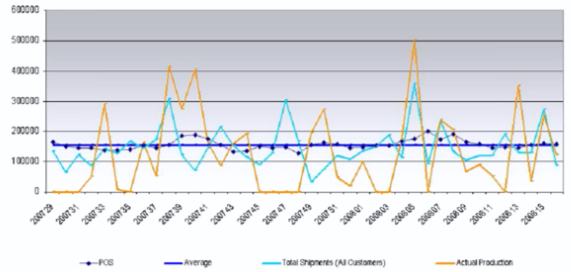


Рівень запасу деяких ресурсів впав до червоної зони. Їх витрата перевищила 66%. Системою опрацьовано скачок попиту і підвищено рівень страхового запасу таких ресурсів. Сформовано автоматичне замовлення

Результати дії ефекту батога



MRP-система з буферами



Класична MRP-система

Висновки

У ході дипломної роботи:

- Досліджено MRP методологію та блок управління запасами класичної MRP-системи.
- Проведено порівняльний аналіз існуючих методів та підходів в управлінні запасами.
- Розроблено програму, у якій замінено застарілі розрахунки блоку MRP на новітні методи управління запасами.
- Зроблено порівняльний аналіз результатів роботи програм під дією ефекту батога.

Перспективи подальших досліджень

- Включати в логіку розрахунку страхового запасу історію коливань.
- Розробити блок прогнозування попиту.

Дякую за увагу!